Cena 13500 zł

DODATEK o ST

tajemnice ATARI

9 (29)/93

PISMO UŻYTKOWNIKÓW ATARI

W NUMERZE:

AB...

Kompresja od środka 3-6
Pisemny DEMO 7-9
Pisemny GRE 11-12

A Strona 6

Joy Transmission 21-23
O magnetofonie... 24

Nalot 18

Digi Duck 16-17
Z pamiętnika łamacza joysticka 19-20

Esmc 13
Komdo 14

Nowości Avalonu 15

Oferujemy dla Twojego Atari (XL/XE)

- **A.D. 2044** cena: 49.000
  - gra przygodowa (na motywach filmu "Seksmisja").
- **ADAX** cena: 49.000
  - komnatówka, misja zwiadowcy na planecie Adax
- » **AURUM** cena: 49.000
  - komnatówka, poszukiwania recepty na produkcję złota.
- **AUTOMAT PERKUSYJNY** cena: 49.000
  - program użytkowy do tworzenia "muzyczek" komputerowych; obszerna instrukcja obsługi.
- » **BARAHIR** cena: 49.000
  - gra przygodowo-tekstowa.
- **CAPTAIN GATHER** cena: 40.000
  - gra logiczna, pierwsze plansze wydają się łatwe ...
- **CHANGE** cena: 40.000
  - coś dla miłośników łamigłówek
- **CONSTELLATION** cena: 40.000
  - gra ekonomiczna (tylko na dyskietce).
- **CHAOS MUSIC COMPOSER** cena: 49.000
  - pakiet zawierający program do komponowania, moduł odtwarzający oraz wyczerpującą instrukcję.
- » **CZASZKI & ELECTRA** cena: 49.000
  - dwie gry zręcznościowe.
- **DAGOBAR** cena: 40.000
  - gra logiczna, kolejne przygody dzielnego Kapitana Gathera.
- **DARKNESS HOUR** cena: 49.000
  - gra komnatowa; poszukiwanie zaklętych dusz rycerzy.
- **DIGI DUCK** cena: 49.000
  - Robbo? Nie! Chodzący telewizor.
- » **GEOMETRIA** cena: 49.000
  - program edukacyjny z zakresu geometrii wykreślnej, obejmuje materiał szkoły podstawowej.
- » **GLOBAL WAR** cena: 49.000
  - gra strategiczna, wojna nuklearna.
- **GOLD HUNTER** cena: 49.000
  - gra przygodowa, poszukiwacz złota w akcji.
- **HANS KLOSS** cena: 49.000
  - gra przygodowo-zręcznościowa, poszukiwania planów tajnej broni w bunkrze Wolfschanze.
- » **HONKY & EASY MONEY** cena 49.000
  - zestaw dwóch gier zręcznościowych.
- **HYDRAULIK+SNOWBALL** cena: 49.000
  - dwie gry zręcznościowe.
- **KAMPANIA WRZEŚNIOWA** cena: 49.000
  - gra strategiczna oparta na historii wojny obronnej 1939r.
- **KERNAW** cena: 49.000
  - gra komnatowa (zmagania mutanta ze smokiem).
- **KLĄTWA** cena: 49.000
  - gra przygodowa, koncepcja i grafika podobna jak w A.D. 2044.
- **KURS FIZYKI** cena: 49.000
  - program edukacyjny, dudnienia, załamanie i odbicie fali, efekt Dopplera.
- **LASERMANIA, ROBBO KONSTRUKTOR** cena: 49.000
  - efektowna gra logiczna oraz program użytkowy pozwalający na tworzenie własnych wersji gry ROBBO.
- **LORIEN'S TOMB** cena: 40.000
  - ekspedycja archeologiczna do grobowca faraona.
- **MAJOR BRONX** cena: 40.000
  - gra zręcznościowo-logiczna.
- **MISJA, FRED** cena: 49.000
  - zmagania komandosa w tajnej MISJI;
  - wędrówka przez pełną niebezpieczeństw krainę.
- **NERON** cena: 49.000
  - opis wewnątrz numeru 4/93.
- **PANTHER** cena: 49.000
  - wielofunkcyjny edytor tekstów pozwalający drukować po polsku na wszystkich popularnych drukarkach.
- **PANTHER + MICROPRINT** cena: 230.000
- **PIEKIEŁKO** cena: 40.000
  - prawdziwie piekielna gra, ale wciągająca.
- **QUICK ASSEMBLER** cena: 69.000
  - zestaw do programowania w języku asemblera: QA - pakiet edytora, asemblera i mini-debugera; BUG HUNTER - nowoczesny debuger całoekranowy; XL FRIEND - edytor tekstów, tabela kodów ASCII; Zestaw procedur bibliotecznych; Podręcznik zawierający krótki kurs programowania.
- **ROBBO** cena: 49.000
  - wspaniała gra zręcznościowo-logiczna.
- **RUCU** cena: 40.000
  - komputerowa kostka RUBIKA
- **SAPER** cena: 49.000
  - gra logiczno-zręcznościowa.
- » **SMUŚ** cena: 49.000
  - gra zręcznościowo-komnatowa.
- » **SPY MASTER** cena: 56.000
  - 'Kloss to przy nim dziecko', 4 gry w jednej!!!
- **SYN BOGA WIATRU** cena: 49.000
  - gra komnatowa (poszukiwacz skarbów w starożytnym mieście Indian)
- **SZPERACZ DYSKOWY** cena: 49.000
  - program użytkowy służący do obsługi szeroko pojętych operacji dyskowych.
- **THE JET ACTION** cena: 40.000
  - kosmiczny RIVER RAID dla łamaczy joysticków.
- **THE LAST GUARDIAN** cena: 49.000
  - ostatni obrońca w akcji - świetna strzelanina.
- **TRIX** cena: 40.000
  - wciągająca gra zręcznościowo-logiczna.
- **U 235** cena: 49.000
  - krótka recenzja: "rozkosz łamania głowy"
- **UCZEŃ CZARNOKSIĘŻNIKA** cena: 49.000
  - gra komnatowa, przygody adepta czarnej magii.
- **UPIÓR** cena: 49.000
  - gra komnatowo-labiryntowa, przygody śmiałka w zamku pełnym niebezpiecznych stworów
- **VICKY** cena: 49.000
  - przygody młodego wikinga w krainie Walhalli.
- **WŁADCY CIEMNOŚCI** cena: 51.000
  - bohater Klątwy znów w akcji!
- » **ZEUS** cena: 49.000
  - poszukiwanie Afrodyty w tajemniczych katakumbach.

Programy na licencji ZEPPELIN GAMES:

(Cena każdej z gier: 49.000)

- **BLINKY'S SCARY SCHOOL**
  - przygody dobrego duszka. Znakomita grafika, dużo efektów specjalnych.
- **CAVERNIA**
  - zbieranie skarbów w kompleksie jaskiń.
- **MIRAX FORCE**
  - kosmiczna walka z eskadrami nieprzyjaciela.
- **NINJA COMMANDO**
  - aby przeżyć, musisz walczyć.
- **SPEED ACE**
  - symulator wyścigu motocyklowego.
- **STACK UP**
  - gra zręcznościowo-logiczna, układanka.
- **ZYBEX**
  - podróż kosmiczna przez niebezpieczne światy.

Programy na licencji K-SOFT:

(Cena każdej z gier: 49.000)

- **ANDROID**
  - gra komnatowa, zmagania androida z najeźdźcami.
- **IMAGINE**
  - przygody maga w krainie snów.
- **MICROX**
  - gra edukacyjna, układanie wzorów związków chemicznych.

PROGRAMY INNYCH FIRM

- **ARTEFAKT PRZODKÓW** cena: 69.000
  - ponad 100 różnorodnych komnat, znakomita grafika, płynna animacja.
- **AUDIO MASTER** cena: 69.000
  - program muzyczny służący do tworzenia i obróbki próbek dźwiękowych.
- » **SAMPLER** cena: 290.000
  - Audio Master + urządzenie do digitalizacji dźwięku.
- **BANG! BANK!** cena: 49.000
  - strzelanina na Dzikim Zachodzie.
- **BATTLE SHIPS** cena: 49.000
  - gra w "statki" z komputerem.
- **BERTYX** cena: 49.000
  - zmagania z wrogim komputerem na planecie Bertyx.
- » **DEIMOS** cena: 55.000
  - wyprawa po legendarny klejnot.
- » **DROP IT** cena: 55.000
  - 3D - Tetris.
- **DUCH, ROBAL** cena: 49.000
  - zestaw dwóch gier labiryntowych.
- **DWIE WIEŻE** cena: 56.000
  - gra 'fantasy'.
- **EUREKA** cena: 49.000
  - stajesz do turnieju o tytuł najmądrzejszego z genialnych.
- **FATUM** cena: 52.000
  - typowa gra przygodowa, mnóstwo strzelania i walki.
- » **FILM EDITOR** cena: 49.000
  - program do tworzenia filmów animowanych.
  - TYLKO NA KASECIE!!! (przepraszamy w imieniu wydawcy).
- **GALLAHAD** cena: 49.000
  - gra komnatowa, walka Tristana z podstępnym Magiem
- **GEOGRAFIA POLSKI** cena: 49.000
  - prosty program edukacyjny (klasa V i VI).
- **GEOGRAFIA ŚWIATA** cena: 49.000
  - program edukacyjny (państwa, stolice, waluty itp.).
- **GUARD** cena: 55.000
  - pilot kosmicznego myśliwca walczy z eskadrami wroga.
- **HELIX** cena: 49.000
  - bohaterski Helix przemierza komnaty starego zamczyska.
- » **HISTORIA POLSKI** cena: 49.000
  - program edukacyjny
- **HUMANOID** cena: 55.000
  - gra zręcznościowo-logiczna.
- **HYDRAULIK** cena: 49.000
  - gra zręcznościowo-logiczna. Niesforna woda wylewa się wszystkimi możliwymi dziurami.
- » **INCYDENT** cena: 55.000
  - podwodna akcja ratunkowa
- **INSPEKTOR** cena: 49.000
  - gra przygodowo-komnatowa.
- **I.Q. MASTER** cena: 49.000
  - gra zręcznościowo-logiczno-edukacyjna.
- **JAFFAR** cena: 55.000
  - gra komnatowa
- **JUMPING JACK** cena: 49.000
  - gra zręcznosciowo - logiczna
- » **KNOCK** cena: 49.000
  - gra logiczno-zręcznościowa.
- **KOLONY** cena: 49.000
  - gra ekonomiczna.
- **KRUCJATA** cena: 55.000
  - gra komnatowa
- **KULT** cena: 51.000
  - pomysł jakby znany (ZYBEX), szybka akcja, dużo strzelania.
- » **KUPIEC** cena: 55.000
  - gra handlowa
- **MAGIA KRYSZTAŁU** cena: 57.000
  - wędrówka maga Tannatosa ku twierdzy demona Syrylaka.
- » **MAGIC DIMENSION** cena: 49.000
  - gra przygodowa.
- **MASTER HEAD** cena: 55.000
  - gra logiczna z kilkunastominutową muzyką.
- **MICROMAN** cena: 49.000
  - wciągająca komnatówka.
- **MIDNIGHT** cena: 55.000
  - gra komnatowa, ratujesz zamek przed zniszczeniem.
- **MIECZE VALDGIRA** cena: 49.000
  - przygodowa gra fantasy z doskonałą muzyką.
- **MIECZE VALDGIRA II** cena: 69.000
- **MONSTRUM** cena: 49.000
  - popularny "robal" w nowej oprawie.
- **MÓZGPROCESOR** cena: 49.000
  - wznowienie znanej gry tekstowej.
- **MUFF, DRUTT** cena: 49.000
  - zestaw dwóch gier zręcznościowych.
- **NEW WORDS** cena: 49.000
  - program edukacyjny, kurs języka angielskiego.
- **OPERATION BLOOD** cena: 49.000
  - krwawe starcie z niezliczoną ilością przeciwników.
- **ORTOGRAFIA** cena: 49.000
  - program edukacyjny, nauka połączona z dobrą zabawą.
- **OUTRIS** cena: 55.000
  - gra logiczna na bazie Tetrisa
- » **PICASSO** cena: 55.000
  - prosty program graficzny
- **PONG** cena: 49.000
  - gra zręcznościowa.
- **PROBLEM JASIA** cena: 49.000
  - puzzle ze zdjęciami popularnych aktorów.
- **PRZEMYTNIK** cena: 49.000
  - gra ekonomiczna.
- **RASZYN** cena: 49.000
  - polska gra strategiczna, bitwa pomiędzy VII korpusem austriackim a wojskami Księstwa Warszawskiego.
- **REVERSE/MISSILE ACTION** cena: 49.000
  - dwie gry logiczno zręcznościowe
- **RIOT** cena: 49.000
  - walki karate w cygańskiej dzielnicy
- **RODERIC** cena: 49.000
  - przygody rycerza w starym zamczysku.
- **SHIP** cena: 49.000
  - przygody cybernoida w statku kosmicznym.
- **SKARBNIK** cena: 49.000
  - gra logiczna.
- **SOS SATURN** cena: 49.000
  - Saturn wzywa Twojej pomocy, gra komnatowa.
- **SOUND TRACKER** cena: 55.000
  - program muzyczny, wzorowany na imienniku z Amigi.
- » **SPECIAL FORCES** cena: 55.000
  - Operation Blood II.
- » **SUPER FORTUNA** cena: 55.000
  - kolejna mutacja "Koła Fortuny".
- **TACTIC** cena: 55.000
  - gra logiczno-zręcznościowa
- **TANKS** cena: 49.000
  - walka czołgów w labiryncie
- » **TARKUS** cena: 49.000
  - gra zręcznościowo-przygodowa
- **THE CONVICTS** cena: 49.000
  - gra labiryntowa (podobna do ROBBO).
- **THINKER** cena: 49.000
  - dwie proste, lecz ciekawe gry logiczna.
- **TRON - Cyki Świetlny** cena: 49.000
  - gra oparta na motywie z filmu TRON
- **TURBICAN** cena: 56.000
  - komnatowa gra logiczna, dziecko Blinky'ego w akcji.
- **T v T** cena: 55.000
  - gra strategiczna dla dwóch osób
- **VIDEOGRAPH** cena: 49.000
  - narzędzie do tworzenia czołówek filmów video
- **WHOOPS** cena: 49.000
  - gra zręcznościowa.
- » **WYPRAWY KUPCA** cena: 49.000
  - gra ekonomiczna.
- » **ZBIR** cena: 49.000
  - gra labiryntowo-przygodowa.

- **ZESTAWY GIER Z TA:**
  1. Fac, QQ, Robal, Dalej Niż Słońce.
  2. Heartlight, Puzzle, Oczko.
  3. Odsiecz, Follow, Klocki, Pustynna Burza.
  4. AQQ, Golf, Pong, Kulki.
  5. The Jet, Piramida, Żaba, Inwazja.
  6. $E=mc^2$, The Moving Cubes, Pillbox, Zjadacz

  Cena zestawu: 38.000, Cena kompletu (1-3 lub 4-6): 95.000

Zobacz także na stronę 25.

Zamówienia na kartach pocztowych, z wyraźnym oznaczeniem rodzaju nośnika (kaseta lub dysk) oraz pełnym adresem zamawiającego prosimy kierować do:

L.K. AVALON
skr. poczt. 46
38-100 Strzyżów

Uregulowanie należności następuje przy odbiorze przesyłki

KOMPRESJA OD ŚRODKA

Drodzy Czytelnicy!

Chyba najważniejszą z wieści, jakie chciałbym Wam przekazać jest to, że już od następnego numeru wracamy do dawnej objętości, czyli trzydziestu dwu stron. Żegnamy się z działem ST i znów cała gazeta będzie traktować o małym ATARI. Jak Wam się to podoba?

Wewnątrz numeru znajdziecie kilka dużych artykułów. Tak się złożyło, że drukujemy je razem. Być może niektórym z Was nie spodobają się takie "kobyły", ale z drugiej strony są tematy, których nie można zmieścić na jednej stronie. Polecam zwłaszcza tekst o kompresji, być może nareszcie przestanie być owianą tajemnicą wiedzą magiczną, dostępną jedynie dla nielicznych wybrańców.

Inny ciekawy temat to transmisja danych pomiędzy ATARI XL i Commodore 64. Wydaje mi się, że dogadanie się choćby przez kabelek będzie krokiem w kierunku zakończenia odwiecznej wojny pomiędzy posiadaczami tych komputerów. Czy to nie jest lepsze?

Spodobały się Wam poprawki do gier – dostaniecie ich więcej. Będziemy je drukować, jeśli i Wy zechcecie podzielić się z nami swoim dorobkiem. Piszcie o wszelkich ułatwieniach, które znaleźliście.

Za miesiąc przedstawimy Wam pakiet programów do obsługi modułu RAM-CART. Myślę, że przyda się Wam przede wszystkim handler, który umożliwi współpracę tego urządzenia z dowolnym DOS-em. Przygotowujemy też drugą część artykułu o prawdziwych programistach, o którą prosiło wielu Czytelników. Oprócz tego będzie porcja zwykłych, comiesięcznych atrakcji. Do zobaczenia.

Jarosław Syrylak

Dość popularny staje się ostatnio problem zmniejszania rozmiarów zbiorów wszelkimi dostępnymi sposobami. Jednym z nich jest:

archiwizacja,

czyli najbardziej ekonomiczny sposób przechowywania nieużywanych danych. Programy archiwizujące używają najróżniejszych metod kompresji począwszy od prymitywnych (*imploding*), a skończywszy na bardzo rozbudowanych (*deflating*). Zadaniem ich wszystkich jest zmniejszenie liczby bitów informacji wejściowej do minimum tak, aby w przyszłości informacja ta mogła zostać odtworzona.

Jaki algorytm zastosować do kompresji jakiego typu zbiorów i na czym polega kompresja oraz jak wygląda to w praktyce?

Na początek bardzo prosta metoda zwana wyszukiwaniem **P**owtarzających **S**ię **E**lementów ciągu (*PSE*). Przykładowo, mając ciąg abbbbccaaabffffffccc możemy zapisać go następująco: a4b2c3ab5f3c. Oznacza to, że a wystąpiło na początku ciągu dokładnie raz, następne było b, które powtórzyło się cztery razy itd.

W rzeczywistości rzecz ma się trochę inaczej. Spróbujmy więc zagęścić na przykład następujący ciąg: `122224444436666`. Otrzymujemy: `14254346`, nieprawdaż? Ale czym odróżnić długość powtarzających się elementów od elementu ciągu. Nie da się? Bzdura! Wprowadźmy dodatkowy bit na początku każdego elementu ciągu wyjściowego wskazujący na to, czy:

- jest to element powtarzający się dokładnie raz (przyjmijmy bit ustawiony),
- jest to wskaźnik liczby powtarzających się elementów (bit skasowany).

Dość opowiastek na dzień dobry. Pora zasiąść przed klawiaturą. Na początek nakarmimy asembler garstką informacji o systemie.

Klocek pierwszy.

```
_opt equ %10010101
_org equ $3000
      org _org
      opt _opt

* -procedury ROM
ciov equ $e456
dschar equ $f2b0
* -system
dosrun equ $a
savmsg equ $64
runad  equ $2e0
initad equ $2e2
memhi  equ $2e5
memlo  equ $2e7
hatabs equ $31a
iocb   equ $340
skctl  equ $d20f

io_com equ iocb+2
io_sta equ iocb+3
io_adr equ iocb+4 (2)
io_len equ iocb+8 (2)
io_mod equ iocb+10
io_aux equ iocb+11

* -stałe
shift  equ $8
chn0   equ $00
chn1   equ $10
chn2   equ $20
putt   equ $9
gett   equ $5
putb   equ $b
getb   equ $7
eol    equ $9b
eof    equ $88

len_text equ $80
len_bufor equ $bc1f-_org
outbit equ 3

* -numery komunikatów
hello_m   equ 0
compres_m equ 1
decomp_m  equ 2
get_m     equ 3
put_m     equ 4
err_m     equ 5
long_m    equ 6

* -strona zerowa
pse_z0 equ $80 (2)
```

Znaczenie większości stałych oraz procedur zostało dokładnie opisane w artykułach z cyklu "Programowanie procesora 6502". Dokładnego omówienia wymagają stałe OUTBIT oraz LEN_BUFOR. Pierwsza z nich wskazuje liczbę bitów reprezentujących długość wskaźnika. Prześledźmy dokładnie przypadek dla OUTBIT równego 3.

Tajemnice ATARI, Pismo użytkowników ATARI XL/XE, skr. poczt. 66, 35-959 Rzeszów 2, tel. (Rzeszów) 62-74-71 wewn. 114.
Redaktor naczelny: Jarosław Syrylak; współpracownicy: Marek Pasek, Barbara Zwarycz, Maciej Mach, Piotr Rak.
Opracowanie merytoryczne: L.K. AVALON; opracowanie graficzne: Danuta Sienkowska. Wydawca: Tajemnice ATARI s.c.
Adres redakcji: Rzeszów, Targowa 1, pok. 106. Druk: RZG Drukarnia w Łańcucie, ul. Mickiewicza 10, tel. 42-96.

Za pomocą trzech bitów możemy przedstawić maksymalną liczbę 111 binarnie, czyli 7 dziesiętnie. Oznacza to, iż wskaźnik będzie zliczał wszystkie powtarzające się elementy, których suma nie przekroczy 7. Następnie przyjmie wartość zero, po czym rozpocznie poszukiwanie nowego podciągu. Stała LEN_BUF wskazuje długość bufora do odczytu pliku. Większość z Was zapyta dlaczego tej długości nie obliczyć w znany już sposób MEMHI – MEMLO? Otóż zależy nam na tym, aby wszystkie wersje archiwizera były kompatybilne (współpracujące). Niemożliwe byłoby odzyskanie długiego pliku skompresowanego przy użyciu jednego DOS–a pod kontrolą innego w przypadku zmniejszenia się liczby wolnych bajtów pamięci.

Ładnie się przywitajmy,

```
main equ *
     ldx #hello_m
     jsr dsp_m
```

i przystąpmy do pracy.

```
code ldx #compres_m
     jsr dsp_m
     ldx #get_m
     jsr get_text
     bpl *+5
```

Jeżeli wystąpił błąd odczytu specyfikacji znaczy to, iż naciśnięto BREAK i czas przejść do dekompresji.

```
     jmp decode
```

Wszystko w porządku, więc zabezpieczmy się przed roztargnionymi programistami,

```
     ldx #chn1
     jsr close
```

a następnie spokojnie odczytajmy zbiór.

```
     lda #4 -odczyt
     jsr open
     bmi code  -wystąpił błąd!
```

W przypadku wystąpienia błędu sygnalizowanego znacznikiem **N** procesora, następuje ponowna prośba o podanie prawidłowej specyfikacji zbioru.

```
* -czytaj plik
     lda <len_bufor
     sta io_len,x
     lda >len_bufor
     sta io_len+1,x
     jsr mcio
* Ustal dlugosc bufora
     bmi code -błąd
     lda io_sta,x
     cmp #eof
     beq isok
```

Gdy plik zmieści się cały, otrzymamy wartość statusu równą $88 (skończyły się dane, plik umieszczony w całości w buforze). W przeciwnym wypadku bufor został zapełniony, ale znacznik końca pliku nie osiągnął wartości krytycznej (został wczytany tylko kawałek pliku). Poinformujmy użytkownika o tym fakcie odpowiednim komunikatem.

```
     jsr close
     ldx #long_m
     jsr dsp_msg
     jmp code
```

Ustalmy rzeczywisty koniec pliku w pamięci oraz fikcyjnie powiększmy go o jeden. Wynika to z ustawienia znacznika flagowego **C** procesora również w przypadku równości zmiennych zawartych w słowach PSE_Z0 oraz END_FILE kontrolowanych w podprogramie CHECK_RG.

```
isok clc
     lda io_len,x
     sta read
     adc bufa
     sta end_file -ustal koniec pliku
     lda io_len+1,x
     sta read+1
     adc bufa+1
     sta end_file+1

     inc end_file
     bne *+5
     inc end_file+1
```

Plik został wczytany, zmienne ustalone, przystąpmy zatem do zapisu wyników kompresji.

```
code_out ldx #chn1
     jsr close
     ldx #put_m -target
     jsr get_text
     bpl *+5 -BREAK zmiana na odczyt
     jmp code
     ldx #chn1
     lda #8  -plik do zapisu
     jsr open
     bmi code_out
```

Ustawmy jeszcze liczniki programu.

```
     jsr reg_zero
```

Do tego miejsca wszystkie opisywane metody kompresji będą takie same. Podobnie podprogramy również będą wyglądały analogicznie.

Wyzerujmy znacznik zakończenia pliku.

```
     lda #0
     sta all_mode
```

Pobierzmy pierwszy bajt.

```
     jsr get_one
     bcc comp_l
```

Jeżeli skończył się plik (ustawiony znacznik **C** procesora) wykonajmy skok do procedury kończącej kompresję.

```
     jmp all_byte
```

Zapiszmy bajt, jaki otrzymaliśmy, jeszcze się przyda.

```
comp_l sta rebyte
```

Wyzerujmy wskaźnik powtórzeń

```
comp_f lda #0
       sta ile
```

i pobierajmy kolejne bajty.

```
comp_d jsr get_one
       bcs all_byte -koniec pliku
```

Nowo odczytany bajt przechowajmy w zmiennej COMP_b

```
       sta comp_b
```

oraz porównajmy go z wartością otrzymaną poprzednio.

```
       cmp rebyte
       bne new_byte -różne bajty
```

W przypadku równości sprawdźmy, czy wskaźnik powtórzeń nie osiągnął już swojej maksymalnej wartości.

```
     lda ile
     cmp pse_bit
```

Równość oznacza zakończenie starej frazy i rozpoczęcie nowej.

```
     beq new_byte
```

Zwiększ licznik powtórzeń.

```
     inc ile
```

Operację odczytu powtarzaj w kółko.

```
     jmp comp_d
```

Otrzymaliśmy nowy bajt lub wskaźnik powtórzeń został przepełniony. Zbadajmy, czy poprzedni bajt powtórzył się przynajmniej raz (ILE<>0).

```
new_byte lda ile
     beq only_one
```

Powtórzył się co najmniej raz. Wyślijmy skasowany bit, o którym była mowa na początku.

```
     clc
     jsr putbit
```

Podobno oszczędność jest zaletą prawdziwych programistów. Wykonajmy skok względny bez używania etykiety. W przypadku pomyślności operacji zapisu bitu przeskoczmy o trzy bajty w przód (omińmy JMP CODE_OUT).

W miejscu *+2+3 zaznaczmy asemblerowi, że ma przeliczyć skok względny od tego adresu, przez dwa bajty skoku względnego (rozkaz BPL oraz jednobajtowy operand) i trzy bajty z następnej lini (jeden bajt rozkazu JMP i dwa bajty adresu CODE_OUT). Sposób ten jest nieco skomplikowany, jednak odciąża asembler.

```
     bpl *+2+3 (bmi code_out)
     jmp code_out
```

Przesuńmy w lewo 8–OUTBIT razy wartość wskaźnika powtórzeń, aby otrzymać znaczące bity na skrajnej lewej pozycji.

```
     lda ile
     ldx #8-outbit
comp_r beq comp_g
     asl @
     dex
     bpl comp_r  (jmp)
```

Wyślijmy OUTBIT bitów.

```
comp_g ldx #outbit
     jsr pt_xbit
     bpl comp_a
     jmp code_out
```

Do wysłania tylko jeden bajt.

```
only_one sec
     jsr putbit
     bpl *+5  (bmi code_out)
     jmp code_out

comp_a lda rebyte
     jsr pt_8bit
     bpl *+5
     jmp code_out
```

Kolejny sposób oszczędzania. Zastosowanie zmiennej COMP_B bezpośrednio w programie.

```
comp_b equ *+1
comp_c lda #0
     jmp comp_l

all_mode equ *+1
all_byte lda #0
     bne all_end
```

Poprzez zastosowanie sprytnej sztuczki polegającej na jednorazowym przej-

ściu przez poniższe rozkazy zabezpieczamy się przed niewysłaniem np. nie dokończonej frazy lub bajtu znajdującego się w zmiennej COMP_B. Drugie dojście do etykiety ALL_BYTE spowoduje skok do ALL_END. Sprawdźcie sami.

```
     inc all_mode
```

W starym bajcie zamieńmy bity. W ten sposób okłamiemy ATARI, że otrzymało nowy bajt różny od poprzedniego (pozwoli nam to na zamknięcie frazy powtarzających się elementów).

```
     lda rebyte
     eor #$ff
     jmp comp_l
```

Dopilnujmy, żeby wysłane zostały wszystkie ważne bity z bufora.

```
all_end lda pbit_cnt
     cmp #8
     beq all_ok
     jsr putbit
     bpl all_end
     jmp code_out
```

Zamknijmy kanał i zacznijmy wszystko od początku.

```
all_ok ldx #chn1
     jsr close
     jmp code_out
```

Dekompresja danych.

Proces dekodowania zapisu programu zostanie opisany w następnym odcinku tego cyklu. Dziś tylko początek.

```
decode equ *
     ldx #decomp_m -decode
     jsr dsp_msg
     ldx #get_m
     jsr get_text
     bpl *+5
* -break zmiana na kompresje
     jmp code
deco_in ldx #chn1
     jsr close
     lda #4 -odczyt
     jsr open
     bmi decode -błąd !!!

     clc
     lda bufa
     adc <len_buf
     sta end_file
     lda bufa+1
     adc >len_buf
     sta end_file+1

     jsr reg_zer
```

Niecierpliwi mogą spróbować własnych sił. Pozostałym życzę odrobiny cierpliwości, zamieszczam jednak wszystkie procedury do odczytu pojedynczego bitu (GETBIT), całego bajtu (GETBYTE) oraz zapisu odkodowanego bajtu (PUT_ONE).

Procedury I/O.

Większość procedur jest już znana. Znaczenia pozostałych na pewno domyślicie się sami.

```
* -wyślij 8 bitów
pt_8bit ldx #8
* -wyślij x bitów
pt_xbit stx pt_xcnt
     sta ptbyte
pt_xcnt equ *+1
pt_cn lda #0
     bne pt_lop
     ldy #1
     rts
pt_lop dec pt_xcn
     rol ptbyte
     jsr putbit
     bpl pt_cn
     rts

* -proc putbit
putbit rol byte_r
     dec pbit_cnt
     beq *+3
     rts
     lda #8
     sta pbit_cnt
* -znowu 8 bitów do wysłania
     inc write
     bne *+5
     inc write+1
byte_r equ *+1
     lda #0
putbyte pha
     lda #putb
put_u ldx #chn1
     sta io_com,x
     lda #0
     sta io_len,x
     sta io_len+1,x
     pla
     jmp mcio
Weź jeden bajt z urządzenia.
getbyte lda #getb
     pha
     bne put_u (jmp)
```

Przy odczycie pojedynczego bitu wygodnie będzie skorzystać z licznika zapisu bitów PBIT_CNT.

```
getbit inc pbit_cnt
     lda pbit_cnt
     cmp #9
     bcs get_g -pobrano już cały bajt
     rol byte_r
     ldx #chn1
     ldy io_sta,x
     rts
get_g lda #0
     sta pbit_cnt
     jsr getbyte
     bpl *+3
     rts
     sta byte_r
     jmp getbit
gt_8bit ldx #8
gt_xbit stx gt_xcnt
     lda #0
     sta gtbyte
gt_xcnt equ *+1
gt_cnt lda #0
     bne gt_lop
     lda gtbyte
     ldy #1
     rts
gt_lop dec gt_xcnt
     jsr getbit
     bpl *+3
     rts
     rol gtbyte
     jmp gt_cnt

put_one jsr check_rg
     bcc *+3
     rts
     sta (pse_z0),y

     jmp get_cd

get_one jsr check_rg
     bcc *+3
     rts
     lda (pse_z0),y

get_cd inc pse_z0
     bne *+4
     inc pse_z0+1
     rts
```

Sprawdź, czy nie został osiągnięty koniec pliku. Odpowiednio ustaw znacznik **C** procesora.

- PSE_Z0>=END_FILE, **C** ustawiony,
- PSE_Z0<END_FILE, **C** skasowany.

Podczas porównania zachowaj stan akumulatora.

```
check_rg pha
     lda pse_z0
     cmp end_file
     lda pse_z0+1
     sbc end_file+1
     pla
     ldy #0
     rts

reg_zero ldx #1
rg_zr lda #0
     sta write,x
     lda bufa,x
     sta pse_z0,x
     dex
     bpl rg_zr
```

Ustalmy maksymalną wartość wskaźnika.

```
     lda #0
     ldx #outbit
rg_ob sec
     rol @
     dex
     bne rg_ob
     sta pse_bit
```

Zaznaczmy, że żaden bit nie został wysłany ani odebrany.

```
     lda #8
     sta pbit_cnt
     rts

* -zamknij kanał
close lda #12
     sta io_com,x
     jsr ciov
     lda #3
     sta skctl -cicho !
     rts

* -otwórz kanał
open sta io_mod,x
     lda #3
     sta io_com,x
* szukaj dwukropka
     ldy #':'
     cpy text+1
     beq seti
     cpy text+2
     beq seti
     lda #0
* ustaw iocb
seti clc
     adc dnma
     sta io_adr,x
     lda #0
     adc dnma+1
     sta io_adr+1,x
     lda skctl
     and #shift
     asl @
     asl @
     asl @
     asl @
     sta io_aux,x
     jsr mcio
     php
     lda bufa
     sta io_adr,x
     lda bufa+1
     sta io_adr+1,x
     lda io_mod,x
     ora #$3
     sta io_com,x
     plp
     rts

* cio z ew. komunikatem
mcio jsr ciov
     bpl iook
     cpy #eof
     beq iook
error ldx #err_m
derr jsr dsp_msg
     ldy #$ff
     rts
iook ldy #1
     rts
```

Nieznacznie zmodyfikowana procedura odczytu tekstu z edytora.

```
get_text jsr dsp_msg
     ldx #chn0
     lda #gett
     sta io_com,x
     lda txta
     sta io_adr,x
     lda txta+1
     sta io_adr+1,x
     lda <len_text
     sta io_len,x
     lda >len_text
     sta io_len+1,x
     jsr ciov
     bmi nofine

     lda text
     cmp #eol
     bne nofine
```

```
     pla
     pla

     jmp (dosrun)

nofine ldy io_sta,x
     rts
```

Procedura wyświetlania komunikatu.

```
dsp_msg equ *
* odszukaj tekst nr x
     ldy #Ø
fmØ  dex
     bmi mout
fmes lda data,y
     iny
     cmp #eol
     bne fmes
     beq fmØ (jmp)
* wypisz tekst
mout txa
     ldx #chnØ
     sta io_len,x
     clc
     tya
     adc dtaa
     sta io_adr,x
     lda #Ø
     sta io_len+1,x
     adc dtaa+1
     sta io_adr+1,x
     lda #putt
     sta io_com,x
     jmp ciov
```

Pomimo, że program nie jest relokowalny, zachowajmy sposób zapisu niektórych stałych – przyzwyczajenia też są ważne!

```
dtaa dta a(data)
dnma dta a(dnam)
txta dta a(text)
bufa dta a(bufor)

* dane
data equ *
     dta c'TA PSE
de/compressor',b(eol)
     dta c'  Compress',b(eol)
     dta c' Decompress',b(eol)
     dta c'Source:',b(eol)
     dta c'Target:',b(eol)
     dta c'I/O error !',b(eol)
     dta c'Buffor full!',b(eol)

dnam dta c'D1:'
text    org *+len_text

pbit_cnt org *+1 -ile bitów
wysłano/odczytano
pse_bit org *+1 -ile bitów wskaźnika

write  org *+2 -ile bajtów wysłano
end_file org *+2 -rzeczywisty koniec
pliku
read   org *+2 -ile bajtów wczytano
ptbyte org *+1 -zapisywany bajt
gtbyte org *+1 -odczytywany bajt
rebyte org *+1 -powtarzający się bajt
ile    org *+1 -ile takich samych
bajtów wystąpiło
bufor  equ *   -bufor odczytu pliku
```

Nieznacznie zmieniony sposób odczytu nazwy oraz numeru urządzenia, z jakiego program został uruchomiony.

```
run_me ldx $2e
     ldy iocb,x
     lda hatabs,y
     sta dnam
     lda $21
     ora #'Ø'
     sta dnam+1
     rts

     org runad
     dta a(main)
     dta a(run_me)
```

Za miesiąc o tym, jak odkodować *PSE* i jak zakodować *Imploding*!

MATHNOID'93

Program FComp służy do porównywania zawartości dwu plików. Jest kilka zastosowań dla takiego programu. Przykładem może być test handlera, kiedy trzeba sprawdzić czy to, co odczytaliśmy jest tym samym, co zostało zapisane.

Po uruchomieniu program przepisze potrzebne dane na szóstą stronę pamięci i od tej pory można porównywać. Aby tego dokonać trzeba przekazać jako parametry dla funkcji USR adresy nazw plików do porównania.

Gdy jeden plik będzie się różnił od drugiego na ekranie zostaną wypisane następujące informacje:

– odległość w bajtach od początku pliku

– wartość szesnastkowa bajtu w pierwszym pliku

– wartość szesnastkowa bajtu w drugim pliku

– znak ASCII bajtu w pierwszym pliku

– znak ASCII bajtu w drugim pliku

W przypadku, gdy pliki będą identyczne, wartość zmiennej, za pomocą której wywołaliśmy funkcję USR będzie wynosić 1. W przypadku błędu, zmienna ta zawierać będzie jego numer.

```
VI  Ø REM *--------------------------*
WO  1 REM : FComp 1.Ø                :
RK  2 REM :  autor: TapeMan          :
JK  3 REM : (c) 1993 Tajemnice ATARI :
VM  4 REM *--------------------------*
GS  1Ø I=1536:TRAP 3Ø
TD  2Ø READ A:POKE I,A:I=I+1:GOTO 2Ø
NL  3Ø ? "Gotowe.":END
QW  11Ø DATA 1Ø4,1Ø4,168,1Ø4,162,32,32
FY  12Ø DATA 138,6,176,87,1Ø4,168,1Ø4,162
EO  13Ø DATA 16,32,138,6,176,85,169,1,141
LX  14Ø DATA 232,6,32,169,6,176,72,173
NT  15Ø DATA 251,6,2Ø5,248,6,24Ø,42,16Ø,4
NA  16Ø DATA 173,23Ø,6,32,113,6,173,231,6
WL  17Ø DATA 32,113,6,173,251,6,16Ø,7,32
QK  18Ø DATA 113,6,173,248,6,16Ø,1Ø,32
YT  19Ø DATA 113,6,162,234,16Ø,6,32,66
AW  2ØØ DATA 198,169,Ø,141,232,6,173,23Ø
WD  21Ø DATA 6,24,1Ø5,1,141,23Ø,6,144,3
YY  22Ø DATA 238,231,6,76,26,6,1Ø4,1Ø4,76
EQ  23Ø DATA 1Ø6,6,172,232,6,132,212,169
NU  24Ø DATA Ø,133,213,96,72,41,15,32,124
NW  25Ø DATA 6,1Ø4,74,74,74,74,24,1Ø5,48
QR  26Ø DATA 2Ø1,58,144,2,1Ø5,6,153,234,6
PV  27Ø DATA 136,96,157,68,3,152,157,69,3
XZ  28Ø DATA 169,3,157,66,3,169,4,157,74
OO  29Ø DATA 3,169,128,157,75,3,32,86,228
AY  3ØØ DATA 48,2,24,96,56,96,162,16,169
UU  31Ø DATA 7,157,66,3,169,251,157,68,3
MV  32Ø DATA 169,6,157,69,3,32,2Ø9,6,144
AJ  33Ø DATA 1,96,162,32,169,7,157,66,3
NF  34Ø DATA 169,248,157,68,3,169,6,157
DN  35Ø DATA 69,3,169,1,157,72,3,169,Ø
BJ  36Ø DATA 157,73,3,32,86,228,192,136
JB  37Ø DATA 24Ø,2,24,96,56,96,Ø,Ø,Ø,Ø,48
IZ  38Ø DATA 48,48,48,58,32,48,48,32,48
OZ  39Ø DATA 48,32,32,27,Ø,32,27,Ø,155
```

(tap)

Piszemy DEMO

Do końca cyklu redagowanego przez nas zostały już tylko dwa odcinki. W dzisiejszym zamieścimy wszystkie procedury potrzebne do uruchomienia drugiej części demonstracji: PLAYER2.ASM, SKOCZEK.ASM, DLI2.ASM i BARY.ASM.

Pierwsza z nich przygotowuje grafikę PMG:

```
Procedure Equ $9FD5

List_mem  Equ %00000110
List_err  Equ %00000101
Code_mem  Equ %00010000
Code_dsk  Equ %00100000

Colpm1s   Equ $02C1
Hpospl    Equ $D001

Pomoc     Equ $0080
Start_pl  Equ $9000

          Opt List_err+Code_dsk
          Org Procedure


          Ldx <Start_pl
          Ldy >Start_pl
          Stx Pomoc
          Sty Pomoc+$01
          Ldy #$00
          Tya
L1        Sta (Pomoc),y
          Iny
          Bne L1
          Inc Pomoc+$01
          Ldx Pomoc+$01
          Cpx #$98
          Bne L1

          Ldx #$A0
          Ldy #$0F
          Stx Hpospl
          Sty Colpm1s
          Rts

          End of file
```

Procedura czyści obszar pamięci PMG (od etykiety Start_pl) oraz ustawia kolor i pozycję pierwszego z graczy. Tego gracza będzie obsługiwała procedura WSKAZNIK.ASM, zamieszczona w poprzednim artykule.

Druga z procedur, SKOCZEK.ASM, w odróżnieniu od PLAYER2.ASM będzie wykonywana co 1/50 sekundy, a jej zadaniem jest ruch pionowy obrazka z pierwszej części demonstracji.

```
Procedure Equ $A10C

List_mem  Equ %00000110
List_err  Equ %00000101
Code_mem  Equ %00010000
Code_dsk  Equ %00100000

Dlptrs    Equ $0230

Pomoc     Equ $008A

          Opt List_err+Code_dsk
          Org Procedure

          Jmp Init_proc

Int_proc  Inc Pomoc
          Ldy Pomoc
          Cpy #$4C
          Bne L1
          Ldy #$00
          Sty Pomoc
L1        Lda Tablica,y
          Sta Dlptrs
          Rts

Init_proc Ldx #$26
          Ldy #$00
L2        Lda Tablica,x
          Sta Tablica+$26,y
          Iny
          Dex
          Bne L2
          Rts


Tablica   Dta B($92),B($92),B($91)
          Dta B($91),B($90),B($90)
          Dta B($8F),B($8E),B($8D)
          Dta B($8B),B($89),B($87)
          Dta B($85),B($83),B($81)
          Dta B($7E),B($7B),B($77)
          Dta B($73),B($6F),B($6B)
          Dta B($67),B($63),B($5E)
          Dta B($59),B($54),B($4F)
          Dta B($49),B($43),B($3C)
          Dta B($36),B($30),B($29)
          Dta B($22),B($1A),B($12)
          Dta B($0B),B($00)

          End of File
```

Wywołanie procedury spowoduje powielenie wartości z tablicy Tablica w odwrotnej kolejności (etykieta Init_proc). Procedura wywoływana jest co 1/50 sekundy od etykiety Int_proc, gdzie z tablicy Tablica pobierana jest kolejna wartość i wpisywana do rejestru Dlptrs. Żeby uzyskać pożądany efekt konieczne jest skonstruowanie odpowiedniej Display List, co robi procedura następna (DLI2.ASM):

```
Procedure Equ $A000

List_mem  Equ %00000110
List_err  Equ %00000101
Code_mem  Equ %00010000
Code_dsk  Equ %00100000

Colpf0s   Equ $02C4
Colpf1s   Equ $02C5
Colpf2s   Equ $02C6
Dlptrs    Equ $0230
Dmactls   Equ $022F

Ekran     Equ $6600
Obrazek   Equ $8000
Pomoc     Equ $0080

          Opt List_err+Code_dsk
          Org Procedure

          Jmp Init_Dl

          Org Procedure+$92

Dlist     Dta B($70),B($70)
          Dta B($4E)
          Dta A(Obrazek+$02)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
          Dta B($0E),B($0E),B($0E)
```

```
            Dta B($ØE),B($ØE),B($ØE)
            Dta B($ØE),B($ØE),B($ØE)
            Dta B($ØE),B($ØE),B($ØE)
            Dta B($ØE),B($ØE),B($ØE)
            Dta B($ØE),B($ØE),B($ØE)
            Dta B($ØE),B($ØE),B($ØE)
            Dta B($ØE),B($ØE),B($ØE)
            Dta B($ØE),B($ØE),B($ØE)
            Dta B($41),A(Dlist)

Init_Dl     Ldx #$ØØ
            Txa
L1          Sta Procedure,x
            Inx
            Cpx #$92
            Bne L1
            Lda #$3E
            Ldx <Procedure
            Ldy >Procedure
            Sta Dmactls
            Stx Dlptrs
            Sty Dlptrs+$Ø1
            Ldx #$62
            Ldy #$86
            Lda #$9C
            Stx ColpfØs
            Sty Colpf1s
            Sta Colpf2s
            Rts

            End of file
```

Właściwy program dla ANTIC'a rozpoczyna się od etykiety Dlist, czyli $92 bajty od początku procedury (dyrektywa Org Procedure+$92). Program (etykieta Init_Dl) zeruje powstałą "dziurę". Jak pamiętamy, każde $00 w programie ANTIC'a tworzy jedną pustą linię na ekranie. Zmieniając wartość Dlptrs zmieniamy ilość pustych linii przed obrazkiem, co powoduje ruch obrazka w pionie.

Po wyzerowaniu fragmentu Display List procedura DLI2.ASM ustawia wektor Dlptrs, wpisuje odpowiednie wartości do Dmactls i rejestrów kolorów.

Ostatnia procedura drugiej części demonstracji to BARY.ASM. Procedura ta wyświetla na ekranie poruszające się, różnokolorowe linie ekranu zwane potocznie "barami":

```
Procedure Equ $A17D

List_mem  Equ %ØØØØØ11Ø
List_err  Equ %ØØØØØ1Ø1
Code_mem  Equ %ØØØ1ØØØØ
Code_dsk  Equ [illegible]

Colbak    Equ [illegible]
Skstat    Equ $D[illegible]
Vcount    Equ $D4[illegible]
Wsync     Equ $D4[illegible]

Tab_rur   Equ $6[illegible]

Pr[illegible]
L2
                ,y

Czysc_[illegible]

L3              Tab_rur,x
                [illegible]
                Cpx #$DF
                Bne L3
                Rts

Wstaw_rury Ldx #$ØØ
L4          Txa
            Pha
            Ldy Poz_rur,x
            Lda Pozycje,y
            Tay
            Inc Poz_rur,x
            Lda Poz_rur,x
            And #$7F
            Sta Poz_rur,x
            Lda Tab_Kolorow,x
            Tax
            Lda #$ØD
            Sta Count
L5          Lda Kolory,x
            Sta Tab_rur,y
            Inx
            Iny
            Dec Count
            Lda Count
            Bne L5
            Pla
            Tax
            Inx
            Cpx #$Ø5
            Bne L4
            Rts

Pokaz_rury Lda Vcount
            Cmp #$Ø8
            Bne Pokaz_rury
            Ldy #$ØØ
L6          Lda Tab_rur,y
            Sta Wsync
            Sta Colbak
            Iny
            Cpy #$DF
            Bne L6
            Rts


Tab_kolorow Dta B($ØØ),B($ØD),B($1A)
            Dta B($27),B($34)

Poz_rur     Dta B($ØØ),B($Ø5),B($ØA)
            Dta B($ØF),B($14)

Count       Dta B($ØØ)

Kolory      Dta B($F2),B($F4),B($F6)
            Dta B($F8),B($FA),B($FC)
            Dta B($FE),B($FC),B($FA)
            Dta B($F8),B($F6),B($F4)
            Dta [illegible]

            Dta [illegible]B($94),B($96)
            Dta [illegible]($9A),B($9C)
            Dta [illegible]9C),B($9A)
            Dta [illegible]6),B($94)
            Dta B[illegible]

            Dta B[illegible]B($34),B($36)
            Dta [illegible],B($3A),B($3C)
            Dta [illegible],B($3C),B($3A)
            Dta B($38),B($36),B($34)
            Dta B($32)

            Dta B($C2),B($C4),B($C6)
            Dta B($C8),B($CA),B($CC)
            Dta B($CE),B($CC),B($CA)
            Dta B($C8),B($C6),B($C4)
            Dta B($C2)

            Dta B($Ø2),B($Ø4),B($Ø6)
            Dta B($Ø8),B($ØA),B($ØC)
            Dta B($ØE),B($ØC),B($ØA)
            Dta B($Ø8),B($Ø6),B($Ø4)
            Dta B($Ø2)

Pozycje     Dta B($Ø1),B($Ø1),B($Ø2)
            Dta B($Ø3),B($Ø4),B($Ø5)
            Dta B($Ø6),B($Ø8),B($ØA)
            Dta B($ØC),B($ØE),B($11)
            Dta B($14),B($17),B($1A)
            Dta B($1D),B($21),B($25)
            Dta B($29),B($2D),B($31)
            Dta B($35),B($3A),B($3E)
            Dta B($43),B($48),B($4D)
            Dta B($51),B($56),B($5B)
```

```
        Dta B($6Ø),B($65),B($6B)
        Dta B($7Ø),B($75),B($7A)
        Dta B($7F),B($83),B($88)
        Dta B($8D),B($92),B($96)
        Dta B($9B),B($9F),B($A3)
        Dta B($A7),B($AB),B($AF)
        Dta B($B3),B($B6),B($B9)
        Dta B($BC),B($BF),B($C2)
        Dta B($C4),B($C6),B($C8)
        Dta B($CA),B($CB),B($CC)
        Dta B($CD),B($CE),B($CF)
        Dta B($CF)

        End of file
```

Główny program składa się z wywołania kilku podprocedur, dopóki nie zostanie naciśnięty klawisz SHIFT. Jeden raz wywoływana jest podprocedura Przepisz, która powiela tablicę pozycji "barów" w odwrotnej kolejności. "Bary" tworzone są w tablicy Tab_rur. W tablicy tej każdy bajt odpowiada za kolor odpowiedniej linii ekranu. Kolejna podprocedura Czysc_tab zeruje tę tablicę, natomiast Wstaw_rury przepisuje odpowiednie wartości do tej tablicy. Podprocedura Wstaw_rury korzysta z dodatkowych tablic: Kolory – tu zamieszczone są wartości kolorów dla poszczególnych "barów". Każdy "bar" zajmuje 13 linii ekranu, czyli zawiera się w 13 bajtach. Każdy blok 13 bajtów w tablicy Kolory określa wygląd jednego z "barów". Pobieranie danych z tej tablicy jest proste dzięki dodatkowej tablicy Tab_kolorów, która określa, z którego miejsca tablicy Kolory pobierać dane dla kolejnych "barów". Tablica Poz_rur zawiera początkowe pozycje "barów", czyli określa, w które miejsce tablicy Tab_rur przepisywane będą kolejne "bary". Tablica ta modyfikowana jest po każdorazowym utworzeniu "barów". Dodatkowy bajt Count służy jako licznik do przepisywania 13 bajów koloru "bara" (etykieta L5). Ostatnia podprocedura Pokaz_rury tablicę Tab_rur do rejestru Colbak, czyli tworzy "bary" na ekranie.

Na zakończenie oglądnijmy efekt działania powyższych procedur. Do uruchomienia drugiej części potrzebne będą następujące zbiory z poprzedniego odcinka: CZYSC.OBJ (czyści pamięć), MUZYKA.OBJ i OBRAZEK.DAT. Oprócz tego zbiór PLAYER.OBJ inicjujący grafikę PMG oraz procedury dzisiejsze: WSKAZNIK.OBJ, PLAYER2.OBJ, DLI2.OBJ, SKOCZEK.OBJ i procedura uruchamiająca RUN.OBJ:

```
Procedure Equ $Ø6ØØ

List_mem  Equ %ØØØØØ11Ø
List_err  Equ %ØØØØØ1Ø1
Code_mem  Equ %ØØØ1ØØØØ
Code_dsk  Equ %ØØ1ØØØØØ

Nmien     Equ $D4ØE
Sizep1    Equ $DØØ9
Skctl     Equ $D2ØF
Vvblk     Equ $Ø222

Bary      Equ $A17D
Dli2      Equ $AØØØ
Music     Equ $74ØØ
Player    Equ $9B5B
Play_2    Equ $9FD5
Reset     Equ $E474
Skoczek   Equ $A1ØC
Wskaznik  Equ $9F95

          Opt List_err+Code_dsk
          Org Procedure

RUN       Ldx Vvblk
          Ldy Vvblk+$Ø1
          Stx Koniec+$Ø1
          Sty Koniec+$Ø2

          Lda #$Ø3
          Sta Skctl
          Ldx <Przerw
          Ldy >Przerw
          Lda #$CØ
          Stx Vvblk
          Sty Vvblk+$Ø1
          Sta Nmien

          Jsr Dli2
          Jsr Player
          Jsr Play_2
          Lda #$Ø3
          Sta Sizep1
          Jsr Skoczek
          Jsr Bary
          Jmp Reset

Przerw    Jsr Music+$Ø1
          Jsr Wskaznik
          Jsr Skoczek+$Ø3
Koniec    Jmp Koniec

          Org $Ø2EØ

          Dta A($Ø6ØØ)

          End of File
```

Zamieszczamy również nieco zmieniony program w Basic'u z poprzedniego odcinka cyklu, który pozwoli na stworzenie samodzielnie działającego zbioru DEMO.COM.

```
 1Ø REM
 2Ø REM PROGRAM LACZACY ZBIORY
 3Ø REM
 4Ø COM NAZWA$(14)
 5Ø OPEN #1,8,Ø,"D:DEMO.COM"
 6Ø FOR L=Ø TO 4Ø:READ D
 7Ø POKE 1536+L,D:NEXT L
 8Ø FOR L=Ø TO 9:READ NAZWA$:? NAZWA$
 9Ø OPEN #2,4,Ø,NAZWA$
 1ØØ D=USR(1536,32,7,14ØØØ)
 11Ø X=USR(1536,16,11,D)
 12Ø CLOSE #2:NEXT L:CLOSE #1:END
 897 REM
 898 REM PROGRAM MASZYNOWY
 899 REM
 9ØØ DATA 1Ø4,1Ø4,1Ø4,17Ø,1Ø4,1Ø4,157
 91Ø DATA 66,3,1Ø4,157,73,3,1Ø4,157
 92Ø DATA 72,3,169,Ø,157,68,3,169,1ØØ
 93Ø DATA 157,69,3,32,86,228,189,72,3
 94Ø DATA 133,212,189,73,3,133,213,96
 997 REM
 998 REM NAZWY ZBIOROW
 999 REM
 1ØØØ DATA D:CZYSC.OBJ
 1Ø1Ø DATA D:MUZYKA.OBJ
 1Ø2Ø DATA D:PLAYER.OBJ
 1Ø3Ø DATA D:WSKAZNIK.OBJ
 1Ø4Ø DATA D:PLAYER2.OBJ
 1Ø5Ø DATA D:DLI2.OBJ
 1Ø6Ø DATA D:SKOCZEK.OBJ
 1Ø7Ø DATA D:BARY.OBJ
 1Ø8Ø DATA D:OBRAZEK.DAT
 1Ø9Ø DATA D:RUN.OBJ
```

Tomasz Bielak
Rafał Bielecki

tajemnice ATARI
skr. poczt. 66
35-959 Rzeszów 2

Zamawiam nagrane na ☐ kasecie ☐ dysku

programy zamieszczone w ------- numerze TA

mój adres:

_____

_____

_____

C&C.soft

USŁUGI KOMPUTEROWE

* ATARI * COMMODORE

– NAPRAWY SPRZĘTU

- KOMPUTERY
- STACJE DYSKÓW
- JOYSTICKI
- MONITORY
- MAGNETOFONY
- INNE

– PROGRAMY NA ZLECENIA

– OPROGRAMOWANIE

35–206 Rzeszów, Poznańska 2a, tel. (0–17)424–00 w. 309

OLBIT sc

Zakład Usług Informatycznych i Handlu

ATARI

800 XL, 800 XE, 65 XE, 130 XE

Instrukcje, literatura, autoryzowane oprogramowanie

Dyskietki, kasety, kartridże, TURBO 2000

Bogata oferta oprogramowania firm polskich

Konkurencyjne ceny!

Katalogi GRATIS

po przysłaniu koperty ze znaczkiem

Tylko wysyłkowo!

00-897 Warszawa 4, skr. poczt. 85
tel. 18-54-09

Redakcja **Tajemnic ATARI** informuje, że posiada jeszcze pewną liczbę egzemplarzy archiwalnych czasopisma. Można je nabywać wysyłkowo wpłacając pieniądze podobnie, jak przy prenumeracie. Oto wykaz dostępnych numerów wraz z cenami:

TA 5/91	Cena: 10 000 zł
TA 7/91	Cena: 10 000 zł
TA 8/91	Cena: 10 000 zł
TA 1/92	Cena: 10 000 zł
TA 5/92	Cena: 10 000 zł
TA 6-7/92	Cena: 14 000 zł
TA 8/92	Cena: 10 000 zł
TA 9/92	Cena: 12 500 zł
TA 10/92	Cena: 12 500 zł
TA 11–12/92	Cena: 13 900 zł
TA 1–2/93	Cena: 12 000 zł
TA 3/93	Cena: 12 000 zł
TA 4/93	Cena: 12 000 zł
TA 5/93	Cena: 13 500 zł
TA 6–7/93	Cena: 13 500 zł
TA 8/93	Cena: 13 500 zł

PrenumeraTA

Liczba egzemplarzy każdego numeru	cena 1 egzemplarza	3-mies.	6-mies.
n=1	12900	38700	77400
n=2	12200	73200	146400
n=3	11600	99900	199800
n=4	11100	133200	266400
n=5 do 9	10600	31800*n	63600*n
n=10 lub więcej	9800	29400*n	58800*n

W celu zamówienia prenumeraty należy wypełnić czytelnie blankiet wpłaty na konto (dostępny na pocztach i w bankach) podając na odwrocie imię, nazwisko i adres zamawiającego oraz liczbę egzemplarzy danego numeru wraz z okresem prenumeraty. Pieniądze prosimy wpłacać na konto:

"Tajemnice Atari" s.c. Rzeszów, ul. Targowa 1/801
Bank Przemysłowo-Handlowy III Oddział Rzeszów
nr 326807-23650-136

Ceny ogłoszeń ramkowych w TA:

- 1 cm^2 12 000
- Cała strona 6 mln. + 100% za kolor
- Ostatnia strona (kolor) 15 mln.

Ogłoszenia drobne:

- każde słowo 5 000

ATARI, COMMODORE, IBM

— Komputery, oprogramowanie oryginalne i licencjonowane, osprzęt dodatkowy.
— Sprzedaż, serwis, porady fachowe.
— Modernizacje sprzętu, Turbo do ATARI.
— Tanie drukarki 9-igłowe już od 1950 tys. (polskie znaki w standardzie Mazovii, Latin 2).

oferują:

- Kraków: ul. Bieżanowska 1 "Bit–Plus"
  ul. Wielicka 57 "PLUS"
  tal/fax 56–01–17
- Rzeszów: ul. Rejtana 33 "Bajt" (DH "Sezam")
  tel. 65–48–82
- Jasło: ul. Rynek 14 "Karpaty"
- Nowy Sącz: ul. Szwedzka 1 "VEGA" (w podwórku)

Redakcja **Tajemnic Atari** informuje o możliwości wysyłkowego nabywania wszystkich programów zawartych w dowolnym numerze pisma, nagranych na kasetę (tylko w transmisji normalnej) lub na dyskietkę. W skład zbioru wchodzą wszystkie listingi, zamieszczone w wybranym przez Zamawiającego numerze Tajemnic.

Cena wersji kasetowej, a także dyskowej wynosi 30.000 zł. W cenę wliczony jest koszt nośnika, nagrania, opakowania oraz opłaty pocztowe. Same listingi są za darmo.
Zainteresowanych ofertą prosimy o dokładne wypełnienie zamówienia (na odwrocie) i po naklejeniu go na kartę pocztową, wysłanie na adres redakcji podany w stopce.

Uwaga! Prosimy o nieprzysyłanie wcześniej należności, będzie ona bowiem regulowana przy odbiorze przesyłki.

REDAKCJA NIE PONOSI ODPOWIEDZIALNOŚCI ZA TREŚĆ ZAMIESZCZANYCH OGŁOSZEŃ.

PISZEMY GRĘ

Witam ponownie w naszym cyklu. W poprzednich odcinkach prosiłem o nadsyłanie listów z uwagami, propozycjami czy pytaniami dotyczącymi tego cyklu i ogólnie pisania gier. Z wszystkich otrzymanych (jednego) wywnioskowałem, że Czytelnikami są w przeważającej większości (1:0) początkujący użytkownicy ATARI, mający problemy nawet z wyświetleniem trójkolorowego sprite'a czy napisaniem Display List'y. Tak więc, aby zadość uczynić danym obietnicom spełnienia życzeń Czytelników, zamiast pisać np. o tym jak uzyskać 21 sprite'ów na ekranie, napiszę po prostu o tym, jak wyświetlić trójkolorowego sprite'a. Aby informacje zawarte w artykule były dla wszystkich zrozumiałe, przedstawię je w formie kolejnych kroków do wykonania.

Krok pierwszy.

Wykonaj takie czynności jak dla zwykłych sprite'ów jednokolorowych: rezerwacja pamięci, ustalenie rozdzielczości itd.

Krok drugi.

Ustaw bit 5 w rejestrze GTICTLS ($26f).

Krok trzeci.

Ustal kolory dla sprite'ów. Pamiętaj, że wynikowy kolor w miejscu nałożenia się punktów z obu sprite'ów jest sumą logiczną (OR) kolorów pojedynczych sprite'ów.

Krok czwarty.

Ustaw oba sprite'y w tym samym miejscu ekranu.

Myślę, że prościej nie można tego wyjaśnić. Dla opornych mam dwie rady: 1. analiza zamieszczonego przykładu, 2. rezygnacja z pisania gier i zajęcie się raczej ich wykorzystywaniem.

```
 opt 21
 org $a000

* --- Constans ---

pm_adr   equ $b000

dma_mod  equ %111010
* Normalny obraz,
* DMA dla graczy,
* rozdzelczosc jednoliniowa,
* DMA dla ANTIC'a.

gtia_mod equ %100001
* Najwyzszy priorytet dla sprite'ow,
* trojkolorowe sprite'y.

pm_mod   equ %10
* Zezwolenie na
* wyswietlanie graczy.

pm_size  equ %00
* Normalna szerokosc
* sprite'a.

col_1    equ %01110100
*              +
col_2    equ %10001000
*              =
*            11111100

height   equ 16
* Wysokosc.

min_x    equ 48
min_y    equ 32
max_x    equ 200
max_y    equ 224-height
* Minimalne i maksymalne
* pozycje na ekranie.

pl0_pos  equ $400
pl1_pos  equ $500
* Przesuniecia wzgledem
* poczatku obszaru
* pamieci sprite'ow.

* --- System -----

dmactl   equ $22f
gtiactl  equ $26f
colpm0   equ $2c0
colpf0   equ $2c4

hpos0    equ $d000
sizep0   equ $d008
pmctl    equ $d01d
pmbase   equ $d407

* --- Program ----

begin    equ *

* Krok pierwszy

 ldy #0
 tya

bgn_1    equ *

 sta pm_adr+$300,y
 sta pm_adr+$400,y
 sta pm_adr+$500,y
 sta pm_adr+$600,y
 sta pm_adr+$700,y
 iny
 bne bgn_1

 lda >pm_adr
 sta pmbase

 lda #pm_size
 sta sizep0
 sta sizep0+1

 lda #dma_mod
 sta dmactl

 lda #pm_mod
 sta pmctl

* Krok drugi

 lda #gtia_mod
 sta gtiactl

* Krok trzeci

 lda #col_1
 sta colpm0
 lda #col_2
 sta colpm0+1


* Poczatek przykladu

 lda #0
 sta colpf0+2
 lda #2
 sta colpf0+4

 lda #min_x
 lda #100
 sta x_plr
 lda #min_y
 lda #100
 sta y_plr
 lda #0
 sta zx
 sta zy

* Glowna petla
* przykladu

loop     equ *
```

```
* Oczekiwanie na
* synchronizacje
* pionowa.
* (Przerwanie VBLANK)

 lda 20
 cmp 20
 beq *-2

* Wywolanie procedur
* obslugi sprite'a.

 jsr clr_plr
 jsr new_pos
 jsr shw_plr

* Skok na poczatek
* petli.

 jmp loop

* Skasowanie sprite'a

clr_plr  equ *

 ldy y_plr
 ldx #0
 txa

clpl_1   equ *

 sta pm_adr+pl0_pos,y
 sta pm_adr+pl1_pos,y
 iny
 inx
 cpx #height
 bcc clpl_1
 rts

* Wyswietlenie sprite'a

shw_plr  equ *

 ldy y_plr
 ldx #0

shpl_1   equ *

 lda shape1,x
 sta pm_adr+pl0_pos,y
 lda shape2,x
 sta pm_adr+pl1_pos,y
 iny
 inx
 cpx #height
 bcc shpl_1
 lda x_plr
 sta hpos0
 sta hpos0+1
 rts

* Zmiana pozycji

new_pos  equ *

* Ustalenie nowej
* pozycji poziomej

 ldx x_plr
 inx
 lda zx
 beq right
 dex
 dex

right    equ *

 stx x_plr

* Sprawdzenie, czy
* ma nastapic zmiana
* kierunku ruchu w
* poziomie...

 lda #$00
 cpx #min_x
 bcs *+4
 lda #$ff

 cpx #max_x
 bcc *+4
 lda #$ff

* ...i ewentualna
* jego zmiana

 eor zx
 sta zx

* Ustalenie nowej
* pozycji pionowej

 ldx y_plr
 inx
 lda zy
 beq down
 dex
 dex

down     equ *

 stx y_plr

* Sprawdzenie, czy
* ma nastapic zmiana
* kierunku ruchu w
* pionie...

 lda #$00
 cpx #min_y
 bcs *+4
 lda #$ff

 cpx #max_y
 bcc *+4
 lda #$ff

* ...i ewentualna
* jego zmiana

 eor zy
 sta zy
 rts

* Zmienne

x_plr    dta b(0)
y_plr    dta b(0)
zx       dta b(0)
zy       dta b(0)

* Definicja pierwszego
* sprite'a

shape1   equ *

 dta b(%00011000)
 dta b(%00111100)
 dta b(%01111000)
 dta b(%01110000)
 dta b(%11100000)
 dta b(%11000000)
 dta b(%11000000)
 dta b(%10000001)
 dta b(%10000001)
 dta b(%00000011)
 dta b(%00000011)
 dta b(%00000111)
 dta b(%00001110)
 dta b(%00011110)
 dta b(%00111100)
 dta b(%00011000)

* Definicja drugiego
* sprite'a

shape2   equ *

 dta b(%00011000)
 dta b(%00111100)
 dta b(%01111110)
 dta b(%01111110)
 dta b(%11111111)
 dta b(%11111111)
 dta b(%11111111)
 dta b(%11111110)
 dta b(%11111110)
 dta b(%11111100)
 dta b(%11111100)
 dta b(%11111000)
 dta b(%01110000)
 dta b(%01100000)
 dta b(%00000000)
 dta b(%00000000)

 opt 7
 end
```

P.S. Ciągle czekam na listy. Jak nic nie przyjdzie, to dalej będę pisał takie bzdury. Proponuję, abyśmy współtworzyli ten cykl na zasadach "Czytelnicy piszą, przyjaciółka odpowiada".

Dariusz "De JET" Żołna

Piszę do Was, bo chciałbym Was prosić, byście umieścili w swoim czasopiśmie informację o klubie, który niedawno założyłem z kolegami. Klub ma zrzeszać atarowców 65XE, 130 XL z magnetofonami w celu wymiany oprogramowania na ten komputer. Do klubu może należeć każdy, kto napisze do nas. Klub nosi nazwę "BOOT".

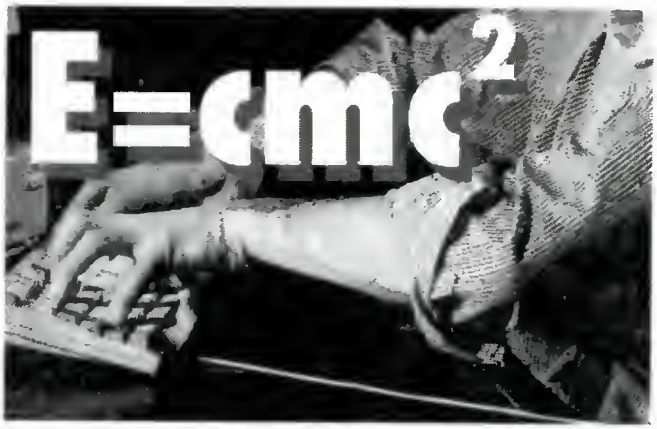
Adres klubu:

"BOOT"
ul. Kolejowa 32/3
68–320 Jasień
woj. zielonogórskie

Od razu napiszę kilka sztuczek do rubryki "Z pamiętnika łamacza joysticka".

- **ADAX** – kombinacja klawiszy **HELP+START+SELECT+OPTION** daje nieskończoną ilość energii, ale skok do wody kończy się śmiercią.
- **RIOT** – wpisanie słów "**zycie to walka**" daje nieskończoną liczbę żyć.
- **AZTEC** – po wgraniu gry należy nacisnąć FIRE i z dodatkowego menu wybrać opcje **S** i **D**.
- **NINJA COMMANDO** – gra wczytuje się etapami; za trzecim razem, gdy taśma się zatrzyma, przed naciśnięciem **START** naciśnij **OPTION**, a zostaniesz nieśmiertelnym.
- **SILENT SERVICE** – klawiszem **?** można jeden raz w grze wypuścić plamę oleju, która zdezorientuje ochronę konwoju.
- **THEATRE EUROPE** – kod do uruchomienia broni jądrowej to **MIDNIGHT SUN**, dla chemicznej – **CAPITAL CITY**.
- **FRED** – wystukanie napisu **LIMEK** daje nieskończoną liczbę kamieni, klawisz **1** zaopatruje w energię i czapeczki, klawisz **2** – w kalosze; kombinację można powtarzać.

R.K. pseudo Gruby



W jednym z poprzednich numerów TA opisywałem procedurę odgrywania muzyczki z poziomu assemblera. Użytkownikom BASIC-a przedstawiłem rozwiązania kilku najczęściej występujących problemów, nie zajmując się sposobami odgrywania muzyczki z poziomu BASIC-a. Uważałem, że odgrywaczka dołączona w zestawie jest wystarczająca. Niestety, nie udało się. Otrzymałem sporo listów z posądzeniem o dyskryminację programistów wysokiego poziomu. Muszę nadrobić tę zaległość.

Załadujmy plik.

Aby to uczynić, należy podać adres początku nazwy pliku jako parametr dla funkcji USR. Aby np. załadować do pamięci muzyczkę ENOLA.CMC i moduł MODULE.REP piszemy taki programik:

```
10 REM Czytaj Pliki
15 DIM NAZWA$(15):SERV=36864
20 NAZWA$="D1:ENOLA.CMC"
25 REM dla magnetofonu NAZWA$="C:"
30 AD=ADR(NAZWA$)
35 MUZ=USR(SERV,AD)
40 REM teraz odtwarzacz
45 NAZWA$="MODULE.REP"
50 REM dla magnetofonu NAZWA$="C:"
55 MOD=USR(SERV,AD)
```

Należy pamiętać, aby przed wykonaniem tego programiku został wykonany program zamieszczony obok, który umieszcza potrzebne dane w pamięci. Po prawidłowym wykonaniu powyższego fragmentu zmienne MUZ i MOD zawierać będą odpowiednio adres muzyczki i modułu odtwarzającego. Adresy te potrzebne są dla odgrywacza.

Włączmy muzyczkę.

Aby rozpocząć odgrywanie muzyczki należy dopisać kolejną linię programiku:

```
60 GRAJ=USR(SERV,MUZ,MOD)
```

Przywróćmy błogą ciszę.

A robi się to tak:

```
65 POKE 764,255
70 IF PEEK(764)=255 THEN 70
80 STOP=USR(SERV)
```

W ten sposób działa program. Rodzaj operacji (ładowanie, odgrywanie, zatrzymanie) zależy od ilości parametrów, które przekazujemy razem z funkcją USR. I tak:

Brak parametrów oznacza, że trzeba wyłączyć muzyczkę, ale uwaga: nie należy używać tej funkcji bez uprzedniego włączenia muzyczki. Jeden parametr oznacza, że trzeba ładować plik o nazwie, która jest przechowywana pod adresem przez ten parametr wskazywanym. Dwa parametry oznaczają, że należy grać muzyczkę, która zawarta jest pod adresem wskazywanym przez pierwszy parametr, korzystając z modułu, który umieszczony jest pod adresem wskazywanym przez parametr drugi.

Zaletą takiego postępowania jest to, że nie trzeba pamiętać wielu adresów do poszczególnych procedur.

```
DU  0 REM *---------------------------*
LY  1 REM : CMC Player 1.3            :
BT  2 REM :  autor: Marek Pasek       :
BM  3 REM : (c) 1993 Tajemnice ATARI  :
DY  4 REM *---------------------------*
VO  10 I=36864:TRAP 20
YL  15 READ A:POKE I,A:I=I+1:GOTO 15
NK  20 ? "Gotowe.":END
GR  110 DATA 104,240,105,201,1,240,126
NP  120 DATA 104,141,61,145,104,141,60
JD  130 DATA 145,104,141,59,145,104,141
IO  140 DATA 58,145,173,60,145,24,105,3
BG  150 DATA 141,78,144,141,84,144,141
JB  160 DATA 124,144,173,61,145,105,0,141
BZ  170 DATA 79,144,141,85,144,141,125
WW  180 DATA 144,173,60,145,24,105,6,141
LD  190 DATA 128,144,173,61,145,105,0,141
TF  200 DATA 129,144,174,58,145,172,59
BC  210 DATA 145,169,112,32,77,144,162,0
LA  220 DATA 138,32,83,144,173,36,2,141
OD  230 DATA 131,144,173,37,2,141,132,144
OF  240 DATA 160,127,162,144,169,7,32,92
RO  250 DATA 228,96,172,131,144,174,132
JJ  260 DATA 144,169,7,32,92,228,169,64
NQ  270 DATA 162,0,32,123,144,96,32,127
NC  280 DATA 144,76,98,228,104,141,69,145
ZR  290 DATA 104,141,68,145,162,16,32,50
QZ  300 DATA 145,169,3,157,66,3,173,68
AI  310 DATA 145,157,68,3,173,69,145,157
XQ  320 DATA 69,3,169,4,157,74,3,169,128
DX  330 DATA 157,75,3,32,86,228,192,1,208
SU  340 DATA 119,169,7,157,66,3,169,62
TH  350 DATA 157,68,3,169,145,157,69,3
RE  360 DATA 169,6,157,72,3,169,0,157,73
NZ  370 DATA 3,32,86,228,173,62,145,201
BC  380 DATA 255,208,7,173,63,145,201,255
WA  390 DATA 240,3,76,43,145,173,64,145
KX  400 DATA 133,212,173,65,145,133,213
MY  410 DATA 169,7,157,66,3,173,64,145
PV  420 DATA 157,68,3,173,65,145,157,69,3
GX  430 DATA 173,66,145,56,237,64,145,141
PW  440 DATA 66,145,173,67,145,237,65,145
BB  450 DATA 141,67,145,173,66,145,24,105
QT  460 DATA 10,157,72,3,173,67,145,105,0
BM  470 DATA 157,73,3,32,86,228,192,136
DH  480 DATA 240,7,192,1,240,3,32,86,245
JA  490 DATA 32,50,145,96,169,12,157,66,3
NG  500 DATA 76,86,228,0,112,0,96,97,116
DM  510 DATA 116,97,99,107,0,0
```

Marek Pasek

W poprzednim numerze TA zabrakło instrumentów do utworu w CMC. Znajdują się one poniżej. Wybaczcie.

```
IJ  1000 REM *---------------------------*
BJ  1002 REM : SUMMER - instrumenty      :
KN  1004 REM :  autor: Marek Pasek       :
ZR  1006 REM : (c) 1993 Tajemnice ATARI  :
JP  1008 REM *---------------------------*
MK  1010 DATA 00ff0000007c3f001223334445
CS  1020 DATA 678bdf01ff0000000ca1f80001
GQ  1030 DATA 2797679bcdef02ff000020707c7
XK  1040 DATA f003333344456 79bdf03ff0000
BR  1050 DATA 000ca1f401112222222223404
MN  1060 DATA ff00000003b50c001122222334
OR  1070 DATA 579d05ff0000000a00c000000
FV  1080 DATA 0000000125 06fffe970003a1f0
JM  1090 DATA 001111223345 68ae07ff008000
VX  1100 DATA 0080fc000011234568acf08ff
KK  1110 DATA 00000000a5040011112233 4568
TW  1120 DATA be09ff00000400b6f003333333
IM  1130 DATA 33333333
```

Oto zaś instrumenty potrzebne do odtworzenia utworu, który drukujemy na następnej stronie.

```
IJ  1000 REM *---------------------------*
GU  1002 REM : Condor - instrumenty      :
KN  1004 REM :  autor: Marek Pasek       :
ZR  1006 REM : (c) 1993 Tajemnice ATARI  :
JP  1008 REM *---------------------------*
TA  1010 DATA 00ff00002000a1f00566677788
AX  1020 DATA 98b96201ff00000100c7f00222
QH  1030 DATA 22444444456c02ff0090000080
RL  1040 DATA f8000000000001802c03ff0000
FO  1050 DATA 300ca1f80333333333456bdf04
XT  1060 DATA ff00000000a6f0043333333333
SB  1070 DATA 333305ff00000000c3f0000002
HR  1080 DATA 2345679bdf06ff00a400008 0ff
YJ  1090 DATA 0000000000012347c07ff008000
NQ  1100 DATA 00c3f000000011234568acf08ff
XC  1110 DATA 0010000080fc000000000 12345a
ZU  1120 DATA cf
```

	ptrn: 00	ptrn: 01	ptrn: 02	ptrn: 03	ptrn: 04	ptrn: 05	ptrn: 06	ptrn: 07	ptrn: 08	ptrn: 09	ptrn: 10
	111: C2# 00	111: C2 01	111: D6 02	111: D4 06	111: A1 04	111: D3 00	111: A1 07	111: A1 05	111: C2 02	111: A4 03	111: A1 01
	2: - --	2: C3 01	2: - --	2: - --	2: - --	2: - --	2: A1 05	2: A1 05	2: C2 02	2: - --	2: A2 01
	3: D2 00	3: C2 01	3: - --	3: D4 06	3: - --	3: D3 00	3: A1 05	3: - --	3: C2 02	3: - --	3: A1 01
	4: - --	4: C3 01	4: - --	4: - --	4: - --	4: - --	4: A1 05	4: A1 05	4: - --	4: - --	4: A2 01
	21: C2# 00	21: C2 01	21: D6 02	21: D4 06	21: - --	21: C3# 02	21: A1 07	21: A1 05	21: C2 02	21: - --	21: A1 01
	2: - --	2: C3 01	2: - --	2: - --	2: - --	2: - --	2: A1 05	2: - --	2: C2 02	2: - --	2: A2 01
	3: D2 00	3: C2 01	3: - --	3: D4 06	3: - --	3: D3 00	3: A1 05	3: A1 05	3: C2 02	3: - --	3: A1 01
	4: - --	4: C3 01	4: - --	4: - --	4: - --	4: - --	4: A1 05	4: - --	4: - --	4: - --	4: A2 01
	31: E2 00	31: C2 01	[illegible]	31: D4 06	31: - --	31: E3 00	31: A1 07	31: A1 05	31: C2 02	31: A4 03	31: A1 01
	[illegible]	2: C3 01	2: - --	2: - --	2: - --	2: - --	2: A1 05	2: A1 05	2: C2 02	2: - --	2: A2 01
	[illegible]	[illegible]	3: - --	3: D4 06	3: - --	3: F3 00	3: A1 05	3: - --	3: C2 02	3: - --	3: A1 01
	4: - --	[illegible]	4: - --	4: - --	4: - --	4: - --	4: A1 05	4: A1 05	4: - --	4: - --	4: A2 01
	41: E2 00	41: F1 01	41: D6 02	41: D4 06	41: - --	41: E3 00	41: A1 07	41: A1 05	41: C2 02	[illegible] --	41: A1 01
	2: - --	2: F2 01	2: - --	2: - --	2: - --	2: - --	2: A1 05	2: - --	2: C2 02	2: - --	2: A2 01
	3: F2 00	3: F1 01	3: - --	3: D4 06	3: - --	3: F3 00	3: A1 05	3: A1 05	3: C2 02	3: - --	3: A1 01
	4: - --	4: F2 01	4: - --	4: - --	4: - --	4: - --	4: A1 05	4: - --	4: - --	4: - --	4: A2 01
	211: G2 00	211: C1 01	211: D6 02	211: D4 06	211: - --	211: G3 00	211: A1 07	211: A1 05	211: C2 02	211: - --	211: A1 01
	2: - --	2: C2 01	2: - --	2: - --	2: - --	2: - --	2: A1 05	2: A1 05	2: C2 02	2: - --	2: A2 01
	3: A2 00	3: C1 01	3: - --	3: D4 06	3: C2 04	3: A3 00	3: C2 05	3: C2 05	3: C2 02	3: C5 03	3: C2 01
	4: - --	4: C2 01	4: - --	4: - --	4: - --	4: - --	4: C2 05	4: C2 05	4: - --	4: - --	4: C3 01
	21: - --	21: C1 01	21: D6 02	21: D4 06	21: - --	21: - --	21: C2 07	21: - --	21: C2 02	21: - --	21: C2 01
	2: - --	2: C2 01	2: - --	2: - --	2: - --	2: - --	2: C2 05	2: C2 05	2: C2 02	2: - --	2: C3 01
	3: - --	3: C1 01	3: - --	3: D4 06	3: - --	3: - --	3: C2 05	3: C2 05	3: C2 02	3: - --	3: C2 01
	4: - --	4: C2 01	4: - --	4: - --	4: - --	4: - --	4: C2 05	4: - --	4: - --	4: - --	4: C3 01
	31: - --	31: C1 01	31: D6 02	31: D4 06	31: - --	31: - --	31: C2 07	31: C2 05	31: C2 02	31: - --	31: C2 01
	2: - --	2: C2 01	2: - --	2: - --	2: - --	2: - --	2: C2 05	2: - --	2: C2 02	2: - --	2: C3 01
	3: - --	3: C1 01	3: - --	3: D4 06	3: - --	3: - --	3: C2 05	3: C2 05	3: C2 02	3: - --	3: C2 01
	4: - --	4: C2 01	4: - --	4: - --	4: - --	4: - --	4: C2 05	4: C2 05	4: - --	4: - --	4: C3 01
	41: - --	41: C1 01	41: D6 02	41: D4 06	41: - --	41: - --	41: C2 07	41: - --	41: C2 02	41: - --	41: C2 01
	2: - --	2: C2 01	2: - --	2: - --	2: - --	2: - --	2: C2 05	2: C2 05	2: C2 02	2: - --	2: C3 01
	3: D3 00	3: C1 01	3: - --	3: D4 06	3: F1 04	3: D4 00	3: F2 05	3: F2 05	3: C2 02	3: F4 03	3: F1 01
	4: - --	4: C2 01	4: - --	4: - --	4: - --	4: - --	4: F2 05	4: F2 05	4: - --	4: - --	4: F2 01
	311: C3 00	311: A1 01	311: D6 02	311: D4 06	311: - --	311: C4 00	311: F2 07	311: - --	311: C2 02	311: - --	311: [illegible] 01
	2: - --	2: A2 01	2: - --	2: - --	2: - --	2: - --	2: F2 05	2: F2 05	2: C2 02	2: - --	[illegible] 01
	3: A2 00	3: A1 01	3: - --	3: D4 06	3: C1 04	3: A3 00	3: C2 05	3: C2 05	3: C2 02	3: C5 03	[illegible]
	4: - --	4: A2 01	4: - --	4: - --	4: - --	4: - --	4: C2 05	4: C2 05	4: - --	4: - --	[illegible]
	[illegible]	21: A1 01	21: D6 02	21: D4 06	21: - --	21: - --	21: C2 07	21: - --	21: C2 02	21: - --	21: C1 01
	[illegible]	2: A2 01	2: - --	2: - --	2: - --	2: - --	2: C2 05	2: C2 05	2: C2 02	2: - --	2: C2 01
	[illegible]	[illegible] A1 01	3: - --	3: D4 06	3: - --	3: - --	3: C2 05	3: C2 05	3: C2 02	3: - --	3: C1 01
	4: - --	[illegible]	4: - --	4: - --	4: - --	4: - --	4: C2 05	4: - --	4: - --	4: - --	4: C2 01
	31: - --	31: A1 01	[illegible] 6 02	31: D4 06	[illegible]	31: - --	31: C2 07	31: C2 05	31: C2 02	31: - --	31: C1 01
	[illegible]	2: A2 01	[illegible]	2: - --	2: - --	2: - --	2: C2 05	2: - --	2: C2 02	2: - --	2: C2 01
	3: - --	3: A1 01	3: - --	3: D4 06	3: - --	3: - --	3: C2 05	3: C2 05	3: C2 02	3: - --	3: C1 01
	4: - --	4: A2 01	4: - --	4: - --	[illegible]	[illegible]	4: C2 05	4: C2 05	4: - --	4: - --	4: C2 01
	41: A2 00	41: A1 01	41: D6 02	41: D4 06	[illegible]	41: [illegible] 00	41: C2 07	41: - --	41: C2 02	41: C5 03	41: C1 01
	2: - --	2: A2 01	2: - --	2: - --	2: - --	2: - --	2: C2 05	2: C2 05	2: C2 02	2: - --	2: C2 01
	3: G2 00	3: - --	3: - --	3: D4 06	3: - --	3: G3 00	3: C2 05	3: C2 05	3: C2 02	3: - --	3: C1 01
	4: - --	4: - --	4: - --	4: - --	4: - --	4: - --	4: C2 05	4: - --	4: - --	4: - --	4: C2 01
	411: F2 00	411: - --	411: D6 02	411: D4 06	411: - --	411: F3 00	411: C2 07	411: C2 05	411: [illegible] 01	411: - --	411: C1 01
	2: - --	2: - --	2: - --	2: - --	2: - --	2: - --	2: C2 05	2: - --	2: C2 02	2: - --	2: C2 01
	3: D2 00	3: - --	3: - --	3: D4 06	3: A1 04	3: D3 00	3: A1 05	3: A1 05	3: C2 02	3: A4 03	3: A1 01
	4: - --	4: - --	4: - --	4: - --	4: - --	4: - --	4: A1 05	4: A1 05	4: - --	4: - --	4: A2 01
	21: - --	21: - --	21: D6 02	21: D4 06	21: - --	21: - --	21: A1 07	21: - --	21: C2 02	21: - --	21: A1 01
	2: - --	2: - --	2: - --	2: - --	2: - --	2: - --	2: A1 05	2: A1 05	2: C2 02	2: - --	2: A2 01
	3: - --	3: - --	3: - --	3: D4 06	3: - --	3: - --	3: A1 05	3: A1 05	3: C2 02	3: - --	3: A1 01
	4: - --	4: - --	4: - --	4: - --	4: - --	4: - --	4: A1 05	4: - --	4: - --	4: - --	4: A2 01
	31: - --	31: - --	31: D6 02	31: D4 06	31: - --	31: - --	31: A1 07	31: A1 05	31: C2 02	31: - --	31: A1 01
	2: - --	2: - --	2: - --	2: - --	2: - --	2: - --	2: A1 05	2: - --	2: C2 02	2: - --	2: A2 [illegible]
	3: - --	3: - --	3: - --	3: D4 06	3: - --	3: - --	3: A1 05	3: A1 05	3: C2 02	3: - --	3: A1 [illegible]
	4: - --	4: - --	4: - --	4: - --	4: - --	4: - --	4: A1 05	4: A1 05	4: - --	4: - --	4: [illegible] 01
	41: - --	41: - --	41: D6 02	41: D4 06	41: - --	41: - --	41: A1 07	41: [illegible]	41: C[illegible]	[illegible]	[illegible] 01
	2: - --	2: - --	2: - --	2: - --	2: - --	2: - --	2: A1 05	2: A1 05	[illegible]	[illegible]	2: A2 01
	3: - --	3: - --	3: - --	3: D4 06	3: - --	3: - --	3: A1 05	3: A1 05	[illegible]	[illegible]	[illegible] A1 01
	4: - --	4: - --	4: - --	4: - --	4: - --	4: - --	4: A1 05	4: - --	4: - --	[illegible]	[illegible] A2 01

			--
			03
02:	[illegible]	9	07
03:	[illegible]	02	06
04:	--	[illegible]	06
05:	01	00	[illegible]
06:	10	05	06
07:	06	00	07
08:	10	05	06
09:	10	02	06
10:	04	07	06

Nowości i Awarie

AURUM

Aurum to kolejna opowieść o mrocznych komnatach i ukrytych skarbach. Jej bohaterem jest wioskowy zawadiaka Stasiek, o którym mówiono, że nie boi się niczego na świecie. Gracz może przekonać się czy to prawda, towarzysząc mu w niebezpiecznej wyprawie. Jego zadanie polegać ma na zgromadzeniu okupu za papirusy, które tajemniczy rabuś skradł zaprzyjaźnionemu ze Staśkiem staremu alchemikowi. Zwoje stanowią dokumentację wieloletnich badań nad produkcją złota. Przezorny staruszek zaszyfrował zapiski, jednak złodziej doskonale zna ich wartość.

Akcja gry toczy się w tajemniczym miejscu, będącym prawdopodobnie podziemiami zapomnianego, leżącego w ruinach zamczyska. Nie należy liczyć na dobre serce i przyjazne uczucia jego mieszkańców. Lata przebywania w mroku uczyniły z nich prawdziwe potwory, przypominające nieco prehistoryczne ptaszyska. Bliskie spotkanie z którymś z nich trudno zaliczyć do przyjemności. Warto jednak podjąć ryzyko, gdyż stawka jest dosyć wysoka, bowiem alchemik obiecał podzielić się wyprodukowanym złotem. Niestety, okup wyznaczony przez rabusia jest naprawdę wygórowany. Złote monety leżą co prawda na korytarzach, lecz do ich zbierania potrzebne jest coś więcej niż dobry wzrok.

Poziom grafiki jest raczej przeciętny, chociaż niektóre rozwiązania są miłe dla oka. Na pochwałę zasługują ptaszyska, pożerające energię gracza. Zostały one ładnie i sugestywnie przedstawione. Szkoda tylko, że autor nie stworzył kilku rodzajów takich (lub innych) "przeszkadzaków". Gra jest dosyć prostą komnatówką, polecam ją wszystkim tym, którzy lubią sobie połazić.

Autorami gry są Dariusz Majer i Artur Wilk, muzykę, budującą nastrój i uatrakcyjniającą zmagania z ponurymi stworami, stworzył Tomasz Licbich.

SMUŚ

Smuś to pechowe smoczątko, które nie słuchało mamusi i wpakowało się w paskudne tarapaty. Maluch nudził się bardzo w rodzinnej pieczarze, więc pewnego dnia, pod nieobecność rodziców postanowił wybrać się na wielką wyprawę. Skończyła się ona niestety w pierwszym napotkanym miasteczku – mieszkający w nim ludzie okazali się bardzo nietowarzyscy. Biedny Smuś został schwytany, a następnie z wielkim hukiem wrzucony do paskudnego lochu. I jak tu teraz wrócić do mamy?

Niegrzeczny smok – junior znalazł się w niecickawym położeniu, a gracz ma za zadanie dopomóc sympatycznemu łobuziakowi wydostać się na światło dzienne. Labirynt komnat, w którym toczy się akcja gry jest miejscem dosyć dziwnym, zamieszkałym przez pełzające tajemnicze stworzenia, podobne nieco do ślimaków. Nie żywią one nadmiernie przyjaznych uczuć do intruza, który błąka się po ich królestwie. Wskazane jest więc unikanie ich, jak również innych istot z podziemnego świata. Smuś niestety nie posiada jeszcze właściwej dorosłym smokom umiejętności ziania ogniem, natomiast znakomicie włada bumerangiem. Przyrząd ten okazał się niesłychanie przydatny w potyczkach z mieszkańcami lochu, ma jednak pewną istotną wadę: aby rzucać nim celnie, należy dobrze się zamachnąć. Niestety, w niektórych miejscach korytarz jest zbyt wąski. Przejścia do kolejnych komnat otwierają się dzięki eksplodującym "guzikom", które porozrzucane są w różnych miejscach lochów. Zbieranie owych niecodziennych kluczy jest niezbędne do posuwania się naprzód.

Scenariusz gry oparty jest na klasycznym pomyśle – główne zadanie gracza polega na przeprowadzeniu uwięzionego bohatera przez podziemia, pełne najdziwniejszych, mniej lub bardziej agresywnych stworzeń. Drogę do wyjścia zagradzają przeszkody, które należy pokonać. Na Smusia czyhają także rozmaite pułapki, zbiorniki z wodą i groźnymi rybami.

Rozwiązania graficzne poszczególnych plansz są niebrzydkie, na uwagę zasługuje zwłaszcza przedstawienie pierwszoplanowej postaci – małego smoka. Jest on ładnie narysowany i budzi dużą sympatię. Animacja w **Smusiu** nie budzi większych zastrzeżeń. Smok nie umie co prawda dobrze latać i jest trochę niezdarny, należy jednak pamiętać o tym, że jest to jeszcze "małolat", uczący się dopiero podstawowych umiejętności, właściwych jego pobratymcom.

Gra przeznaczona jest dla wszystkich miłośników "platformówek", pogromców potworów, a także dla sympatyków smoczego rodu. **Smusia** napisali Maciej Stefański i Tomasz Palka, autorem muzyki jest Tomasz Liebich.

DIGI DUCK

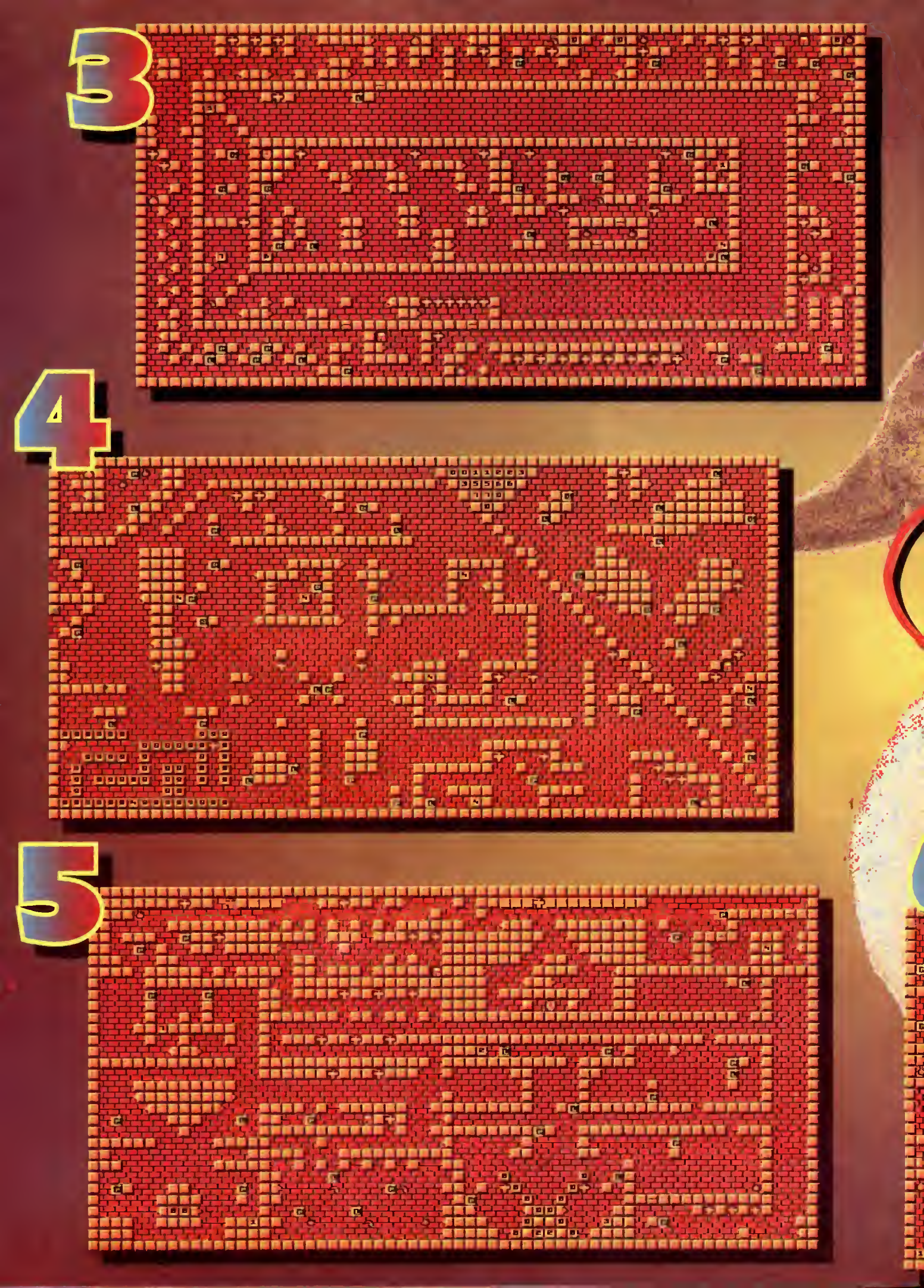
Digi Duck to gra doskonale znana wielu miłośnikom "zręcznościówek" i "labiryntówek". Przenosi nas ona do dziwnego i tajemniczego świata, w którym obdarzony osobowością, inteligentny telewizor walczy o przetrwanie wśród najbardziej podstępnych pułapek. Inwencja autora – Dariusza Żołny wydaje się tu nie mieć granic. Zadanie, jakie postawił przed graczem, wymaga wielu umiejętności. Ważna jest sprawność w posługiwaniu się joystickiem – nawet najbardziej zaawansowani niszczycicle tego urządzenia mogą mieć tu pewne kłopoty. Trzeba bowiem uciekać przed czymś w rodzaju wściekłej kaczki, zręcznie unikać kapiącej z kranu śmiercionośnej substancji, a jednocześnie zbierać bomby, dyskietki, które otwierają przejścia do dalszych komnat oraz rozrzucone tu i ówdzie cyfry. Jakby komuś było jeszcze mało, może spróbować swej zręczności, unikając promienia podstępnie umieszczonego laserowego działa.

Nie są to bynajmniej wszystkie zadania stojące przed graczem – wściekłą kaczkę można ustrzelić, jeżeli bardzo da się we znaki, łatwo niestety również samemu zostać ustrzelonym (koniec żywota symbolizuje ukazujący się w miejsce telewizora gustowny nagrobek), a jeśli nawet nie zdoła unicestwić dzielnego telewizorka, to zawsze można utknąć na dobre przy układaniu równania matematycznego ze znalezionych na planszy cyfr. Oprócz sprawnego posługiwania się joystickiem, trzeba również umiejętnie wykorzystywać intelekt. Mile widziana jest także cierpliwość, bez niej gracz daleko nie zajdzie.

Digi Duck to gra o dość wysokim poziomie trudności, a jej urozmaicona oprawa graficzna i pomysłowy układ plansz zasługują na uznanie. Znaczny wpływ na podniesienie atrakcyjności zabawy ma również wpadająca w ucho dynamiczna muzyka, doskonale pasująca do dramatycznych, ekranowych zmagań. Polecam grę bardziej doświadczonym i trudno zrażającym się amatorom komputerowych wyzwań.

Gracz napotyka nie tylko na utrudniające mu życie pułapki. W niektórych miejscach znajdują się jajka–niespodzianki, które warto zbierać. Ukryte w nich przedmioty i "bonusy" mogą okazać się bardzo przydatne. Przykładowo, z takiego jajeczka może wykluć się... dodatkowy telewizor! Niestety, dojście do takich ułatwień nie jest proste. Na każdą nagrodę gracz musi ciężko zapracować i poświęcić jej zdobyciu wiele czasu. Warto jednak trochę się postarać, gdyż satysfakcja z ukończenia tej niezwykłej gry jest naprawdę niemała. Zapewniam, że wszystkie plansze można przejść, należy jednak trochę wysilić umysł i nie poddawać się zbyt łatwo. Świat należy do upartych.

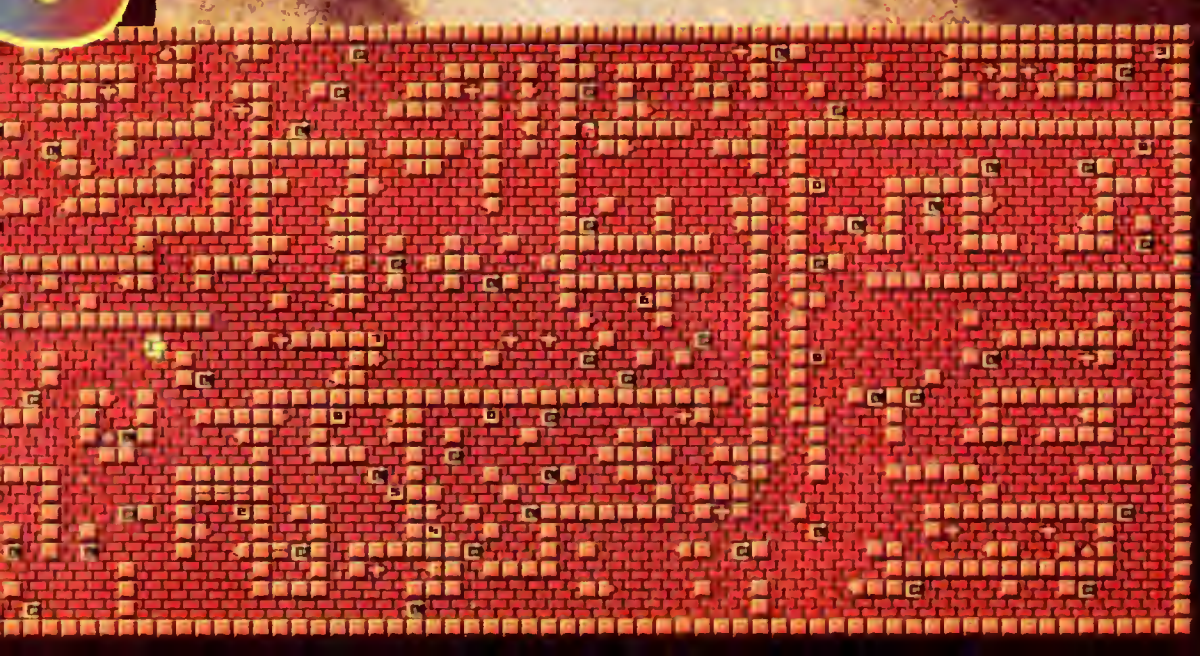
Basta


3
4
5

DIGI DUCK

Równania dla każdego z poziomów:

I. $8x2-7x3+5=0$
II. $2+3+4+5-6=8$
III. $9x8/4-7-6=5$
IV. $4x4-4-4-4=4$
V. $7-1-4x3+8=2$
VI. $7+6-8x2+3=0$
VII. $9/2+7/2-5=0$
VIII. $3/2+9-5/2=8$



NALOT

Kolejny raz zajmiemy się tematyką lotniczą. Prezentowana poniżej gra jest najprostszym symulatorem lotu. Jakim samolotem? A od czego wyobraźnia?

Listing należy przetworzyć przy pomocy *Zgrywusa+*.

```
IJ  1000 REM *----------------------------*
CA  1002 REM : Nalot                       :
JV  1004 REM :  autor: Dariusz Rzepka      :
ZR  1006 REM : (c) 1993 Tajemnice ATARI :
JP  1008 REM *----------------------------*
CT  1010 DATA ffff00904596ad449685972007
QW  1020 DATA 92a9038d0fd2207c9420cb92ad
PE  1030 DATA fc02c91cd0034cf292a9ff8dfc
UJ  1040 DATA 02a003a93d201a93a58285a0a5
OG  1050 DATA 8385a120ec91a9b4a000849284
OC  1060 DATA 9384958496201a93a5828588a5
ZA  1070 DATA 838589a95fa003201a93a58285
BW  1080 DATA 8aa583858ba990a004918888d0
TW  1090 DATA fba991918aa9008591a986858e
GI  1100 DATA a995858fa9ed8d06d28d07d2a9
WN  1110 DATA 29a003201a93a000b18a498038
SU  1120 DATA e910a8c008b022ad0ad2d18eb0
JH  1130 DATA f9859018a591e6916582858290
GD  1140 DATA 02e683204e93a591c916f0084c
ED  1150 DATA 7390a0084c9090a91fa001201a
HQ  1160 DATA 93ad4196a0019182ad4296c891
RM  1170 DATA 82adfc02c91cf05ea592c593f0
EG  1180 DATA 63a9008594854d204a95a583c5
BU  1190 DATA a1f057a003b182f057cd4096f0
UB  1200 DATA 4fa013ad42969182a002ad4196
VG  1210 DATA 918288a9009182204a95a014b1
TW  1220 DATA 82cd4096f053ad4296918288ad
NG  1230 DATA 41969182a002a900918218a912
GO  1240 DATA 658285829002e6834ccb90a9ff
ZR  1250 DATA 8dfc0220f2924c00904c6a934c
DE  1260 DATA b4914c7791a003ad4296918288
IN  1270 DATA ad4196918288a900918218a901
DR  1280 DATA 658285829002e6834ccb90208f
ER  1290 DATA 91a900a0139182c89182a00291
PK  1300 DATA 824c8691208f91a001a9009182
PQ  1310 DATA c89182c8918220849420f2924c
CL  1320 DATA 0090a9328d00d2a9c88d02d2a9
FA  1330 DATA 8f8d01d28d03d2a2ffa0ffeaea
PH  1340 DATA 88eaeaead0f8cad0f3604ce790
PC  1350 DATA a582c5a0d0f7a000b18ac999f0
VD  1360 DATA 13aae88a918aa9008591a00291
SL  1370 DATA 828891824c7390a991918aa900
SR  1380 DATA 8591a0029182889182ad459685
RU  1390 DATA 974c7390a000a9f8201a93a20e
BX  1400 DATA a017a9009182881 0fba9282025
BD  1410 DATA 93ca10ef60a97fa000201a9320
KY  1420 DATA 4393a212a019b1829186881 0f9
MR  1430 DATA a928202593a91a202f93a91a20
SA  1440 DATA 3993ca10e520ec91a000a98320
XD  1450 DATA 1a93a9d19182a04c9182a9c5a0
OZ  1460 DATA 659182a011918288a9d2918288
UN  1470 DATA d0fba9fca0289182a0399182a9
SE  1480 DATA 4c202593a018a9d2918288d0fb
JW  1490 DATA a004a9d89182a0159182a200a9
IH  1500 DATA 282025 93a000bd8f959182a019
LY  1510 DATA 9182e8e010d0eba019a9c39182
ZW  1520 DATA 88a9d2918288d0fba000a9ac20
ZY  1530 DATA 1a93a200bd9f958db592bda995
RQ  1540 DATA 8dbd92bdb3958dc192a000b9bc
QT  1550 DATA 959182c8c010d0f6a9692025 93
CH  1560 DATA e8e00ad0d860a9ff8dfc02adfc
JT  1570 DATA 02c9ffd01aad0fd22908f013ad
EX  1580 DATA 1fd0c906f00cad8402f007a900
ZO  1590 DATA 854d4cd09260a97fa000201a93
VY  1600 DATA 204393a212a019b18691828810
NT  1610 DATA f9a928202593a91a202f93a91a
MV  1620 DATA 203993ca10e560186558858298
UA  1630 DATA 6559858360186582858290 02e6
HB  1640 DATA 836018658485849002e6856018
PH  1650 DATA 658685869002e68760a9468586
VV  1660 DATA a9968587a20160a590d00160ad
AA  1670 DATA 4096a0009182c690a58238e928
RJ  1680 DATA 8582b002c6834c4e93adfc02c9
RJ  1690 DATA 21f016ad0fd22908f02aad8402
QN  1700 DATA f00aad7802c907f01e4cd890a9
TO  1710 DATA ff8dfc02a592f0034cd890a582
KB  1720 DATA 858ca583858da90585924cd890
YH  1730 DATA a9ff8dfc02a593f0034cd890a5
JK  1740 DATA 82858ca583858da90585934cd8
NR  1750 DATA 9088b188c990f018aaca8a9188
CN  1760 DATA c8b188aae8e8e8e8e8e8e88a91
MQ  1770 DATA 88e6954cf893a90085934c6c94
OI  1780 DATA a595d011a003b188c99330ceaa
XC  1790 DATA cacaca8a9188e695a9a08d00d2
KS  1800 DATA a9ef8d01d218a9 01658c858c90
YB  1810 DATA 02e68da002b18cf01dcd4096f0
XC  1820 DATA 3b88b18ccd4296f004a900918c
ME  1830 DATA a900859385952084944c6c94ad
UM  1840 DATA 4396918c88b18ccd4296f004a9
MY  1850 DATA 00918cc693f0034c6c94c8918c
UV  1860 DATA c6952084944c6c94a900918c88
AN  1870 DATA b18ccd4296f004a900918ca900
ZP  1880 DATA 8593859520849 4a594d007a901
JW  1890 DATA 85944c5595a900085946 0a9008d
IW  1900 DATA 06d28d07d2a9008d00d28d01d2
KS  1910 DATA 8d02d28d03d260a596d00aa974
JL  1920 DATA 8d00d2a9ef8d01d218a901658c
OT  1930 DATA 858c9002e68da028b18cf01ecd
EW  1940 DATA 4096f039a000b18ccd4196f004
UE  1950 DATA a900918ca900859285962084 94
ZR  1960 DATA 4c6c94ad4396918ca000b18ccd
TP  1970 DATA 4196f004a900918ca92718658c
DV  1980 DATA 858c9002e68d4c6c94e696a9f6
RZ  1990 DATA 8d00d2a9918d02d2a96f8d01d2
HC  2000 DATA 8d03d2206795a000b18ccd4196
RR  2010 DATA f004a9009 18ca900a028918c18
OD  2020 DATA a927658c858c9002e68dc692f0
JU  2030 DATA 0ca028b18ccd4096f00d4c3895
AZ  2040 DATA 208494a90085964c6c94a497a6
QQ  2050 DATA ffcad0fd88d0f86020 3f95a593
EL  2060 DATA d013a592d00c203f95a593d008
YQ  2070 DATA a592d001604c93944ce393a004
EG  2080 DATA b188c999f006aae88a918860a9
BN  2090 DATA 90918888b188c999f0f5aae88a
KD  2100 DATA 9188600006070 8090a0b0c0dfc
MD  2110 DATA fcd4d480aea1acafb480d4d4fc
JM  2120 DATA fcda00101e2b384559606e7c10
EI  2130 DATA 1e2b384559606e7c84691a1b1b
BF  2140 DATA 6842464245808080b3e3eff2e5
US  2150 DATA 9a9090909080808049550049 55
NW  2160 DATA 004f0000554f495555c94b59c9
TX  2170 DATA 4b595900d94ccf00d9590059cb
BY  2180 DATA 55595900d900d900d959005959
LQ  2190 DATA 0059cb554bcbc900d9246c6100
PM  2200 DATA 0234616a656d6e6963 00213421
CD  2210 DATA 3229026e61706973616c246172
RG  2220 DATA 6975737a00327a65706b610863
XJ  2230 DATA 09001118 0e1011 0e11191913b0
PC  2240 DATA effae9efed8090031d1e543a32
CE  2250 DATA e002e1020090
```

Dariusz Rzepka

Z pamiętnika łamacza joysticka

Cześć, to znowu ja! Opiszę dla Was dwie gry, a mianowicie "Space Lobsters" i "Space Shuttle". Jak widzicie, dzisiaj będzie kosmicznie. A więc do dzieła:

"Space Lobsters" czyli "Kosmiczne homary" to gra komnatowa, dość rozbudowana, bo posiada około 150 komnat. Ale oto historyjka, która wprowadzi Was w jej klimat. Kapitan Crumble (znany być może z innej gry) zatrzymał swój statek gwiezdny MARK 4 przy olbrzymim dryfującym kadłubie porzuconej kolonii statków, która przemieszczała się od wieków w międzygalaktycznej otchłani. Kolos ten umieszczany był zwykle na mapach, lecz nie stanowił obiektu zainteresowania, podobnie jak setki innych wraków dryfujących gdzieś w galaktycznej przestrzeni. Odzyskanie kolosa uważane było za niepraktyczne, biorąc pod uwagę niebezpieczeństwa płynące z wciąż jeszcze aktywnego, lecz znajdującego się w stanie krytycznym napędu wraku. Teraz sprawy wyglądały inaczej – radary wykryły, iż uszkodzony statek kosmiczny niosący na swoim pokładzie śmiertelnie rannego Salmondicusa Giganticusa Labsteriusa dostał się w pobliże owego kolosa i ostatkiem energii nadał sygnał S.O.S. Wiadomość ta została odebrana i rozszyfrowana w Ziemskiej Bazie Dowodzenia. Ludzkość i "kosmiczne homary" od dawien dawna uwikłane były w serię ciężkich, nierozstrzygniętych potyczek, mających na celu zawładnięcie galaktyką...

Nie znamy wroga – narzekał generał Reews – *nie znamy jego kodu genetycznego, jego cyklu rozrodczego, ani też ograniczeń jego IQ. Jedyną rzeczą, jaką o nich wiemy jest to, że nienawidzą ludzkości. Nigdy nie udało nam się schwytać ani jednego żywego przedstawiciela tej rasy. Ale oto nadarzyła się okazja, której nie wolno zmarnować...*

Kapitan Crumble otrzymał zadanie odzyskania statku "homara" wraz z nim samym zanim dotrze tam odsiecz z wrogiej bazy. Zatrzymał się obok wraku i ostrożnie, z napiętą uwagą posuwał się po korytarzach dryfującego statku, wiedząc że gdzieś czeka ranny, być może nawet umierający, ale wciąż groźny przeciwnik. Na swoje nieszczęście kapitan nie wiedział, że homar jest rodzaju żeńskiego i właśnie zakończył składanie jajek. Tak więc korytarze kolosa zapełnione były setkami młodych "homarów" oraz ich "niańkami", zwanymi Robo–droidami. Kapitan Crumble musi pokonać "homary" i innych strażników statku przy pomocy swojego pistoletu plazmowego. Ma on ograniczoną ilość amunicji i musi uzyskać punkty na dalszy zakup magazynków (cena 200 pkt za 50 pocisków). Crumble musi zlokalizować terminale komputerowe, aby rozszyfrować 14 kodów, pozwalających na skorzystanie ze statku.

Na górze ekranu widzimy Crumble'a w tunelu, a poniżej monitor, który jest ekranem komputera. Zobaczymy na nim informacje, gdy podejdziemy do odnalezionego terminalu. Pod monitorem znajduje się ramka z informacją o ilości rozszyfrowanych kodów, obok ilość punktów zdobytych przez Crumble'a, ilość amunicji oraz ilość tlenu w butlach, a także ilość żyć kapitana.

Teraz może "klawiszologia":

- joystick w prawo lub w lewo – poruszanie się w tych kierunkach;
- joystick w górę lub w dół (przy teleporcie) – przemieszczanie się na wyższy lub niższy poziom;
- FIRE – strzał w kierunku, w którym jest odwrócony Crumble;
- klawisz 1 – rozszyfrowanie kodu, jest to możliwe tylko na monitorze;
- klawisz 2 – ujawnianie kodów;
- klawisz 3 – kupowanie amunicji do pistoletu plazmowego (minimum 200 pkt);
- klawisz 4 – powrót do gry;
- OPTION – pauza;

Korzystanie z terminalu komputerowego:

–podejdź do ekranu,
–pchnij joystick do przodu,
–wybierz opcję.

Gra jest dość trudna ze względu na wielość pomieszczeń oraz powolne reagowanie Crumble'a, lecz dostarczy niemało radości tym, którzy opanują joystick do perfekcji.

Homary są smaczne, ale i Ty nie daj się zjeść!

"Space Shuttle" jest testem Twoich umiejętności pilotażu, a wszystkie używane tu urządzenia i wykonywane manewry mają miejsce w prawdziwej misji. Jesteś pilotem promu kosmicznego podczas 101 wyprawy zorganizowanej w ramach Programu Transportu Kosmicznego.

Masz za zadanie:

- wystartować,
- osiągnąć orbitę 210 mil kosmicznych,
- odnaleźć i załadować na prom satelitę, w tak krótkim jak to tylko możliwe czasie, zużywając przy tym jak najmniej paliwa,
- bezpiecznie powrócić na Ziemię.

Lista używanych urządzeń i klawiszy:

- wybór poziomu trudności – **SELECT**,
- klawisz odliczania czasu do startu naciska się po ustaleniu poziomu gry – **L**,
- włączenie/wyłączenie głównych silników promu – **E**,
- drzwi ładowni (*cargo door*) – **C**,
- dodatkowe informacje – **spacja**,
- przerwa w grze – **ESC**.

Joystick – kontrola lotu:

- góra lub dół – ustawia współrzędną X,
- lewo lub prawo – ustala współrzędną Y,
- FIRE i joystick góra lub dół ustala współrzędną Z.

Stopnie trudności:

- **Autosymulator** – nie musisz używać żadnych urządzeń kontrolnych. Od momentu, kiedy poruszysz joystickiem, przejmiesz pełną kontrolę nad wahadłowcem. Używaj joysticka, aby doprowadzić do spotkania z satelitą.
- **Symulator** – możesz używać wszystkich urządzeń tak, jakby to był prawdziwy lot, jednak nie tracisz paliwa, dzięki czemu czas twojego lotu jest nieograniczony. Pokładowy komputer śledzi Twoje ruchy i kiedy popełnisz błąd sygnalizuje Ci to sygnałem dźwiękowym.
- **Pełna misja** – wszystko dzieje się tak, jakby był to przwdziwy lot.

Uruchomienie:

Twoim zadaniem jest uruchomienie promu i osiągnięcie jak najprędzej wysokości 210 mil. Od momentu uruchomienia promu przejdziesz przez trzy fazy:

1. SRB – motory pomocnicze,

2. prom osiąga maksymalne przyśpieszenie,

3. odrzucenie silników.

Litera **X** oznacza *MECO* (moment wyłączenia silników głównych). Zauważysz dwie poziome linie na tablicy kontrolnej. Litera **T** – oznacza dysze, **L** – komputer pokładowy. Uważaj, aby były one równo pod sobą (kontroluj, naciskając FIRE). Platformę opuścisz dopiero prze MET + 3. Patrz uważnie na plan trajektorii. Jeżeli zdarzy Ci się zmienić trajektorię, komputer zasygnalizuje to dźwiękiem. Żółty błysk na wysokości 26 mil oznacza odrzucenie silników pomocniczych. Puste zbiorniki spadną do Oceanu Indyjskiego.

Kolejność działań przy starcie:

- ustaw poziom (1, 2 lub 3),

- start – poczekaj około 8 sekund, aby wszystkie systemy zostały uaktywnione,
- kiedy pokaże się zegar, naciśnij klawisz **L**, aby rozpocząć odliczanie,
- po rozpoczęciu odliczania naciśnij **E**, aby włączyć silniki główne,
- naciśnij przycisk FIRE przy MET – 004, aby zadziałały silniki główne,
- cały czas patrz na plan trajektorii i kontroluj przy pomocy joysticka,
- na wysokości około 200 mil naciśnij **E**, aby wyłączyć silniki główne.

Uwaga: jeśli wyłączysz silniki na wysokości poniżej 195 mil. prom będzie wracał na Ziemię.

Stabilizowanie orbity:

Ustal orbitę przez otwarcie ładowni (klawisz **C**) i ustaw pozycję promu tak, aby mieć kontakt wizualny z Ziemią (jeśli nie otworzysz drzwi ładowni, usłyszysz sygnał alarmowy, masz wtedy około 30 sekund na otworzenie drzwi; jeśli tego nie zrobisz, misja zostanie przerwana). Ustaw dziób promu na dół (PITCH – 28).

Kolejność przy stabilizowaniu orbity:

- naciśnij **C** – otwieranie drzwi,
- naciśnij **R** – uaktywnienie OMS,
- joystick góra lub dół – ustawienie PITCH.

Otworzenie drzwi i ustalenie PITCH muszą być wukonane jako pierwsze, przed innymi operacjami kosmicznymi.

Załadowanie satelity:

Wyrównaj szybkośc i pozycję (współrzędne X, Y, Z) promu i spotkaj się z satelitą (*RANDEZVOUS*). Są dwa różne sposoby manewrowania na orbicie. Dla ważniejszych manewrów (od 30 mil) może być użyty system manewrowania na orbicie OMS. Ten system powinieneś dobrze przestudiować. Jeżeli za pierwszym razem nie uda Ci się ustalić orbity, użyj RCS (System Kontrolowanej Reakcji). Włącza on silniki promu w części dziobowej i bocznej, a następnie przemieszcza prom według trzech współrzędnych. Obserwuj zapalanie się kontrolki sygnalizującej włączenie RCS.

W celu użycia OMS naciśnij **R**, aby zaktywizować ROT (silniki rotacyjne). Joystickiem lewo lub prawo, aby poprawić YAW, góra lub dół by poprawić PITCH. Naciśnięcie FIRE zwiększy moc silników.

W celu użycia RCS naciśnij **T**, aby zaktywizować TRN. Joy góra lub dół, aby poprawić szybkość, lewo lub prawo, aby poprawić współrzędną Y, góra lub dół z naciśniętym FIRE, aby poprawić współrzędną Z (wysokość).

Szybkość jest tak samo ważna jak pozycja. Nie pozwól, aby spadła poniżej 17.0 machów, a wysokość poniżej 195 mil, gdyż możesz zderzyć się z atmosferą. Poprzez regulację współrzędnej X możesz oddziaływać na szybkość. Aby dopędzić satelitę, kiedy jest ona przed Tobą (X dodatnie), Twoja szybkośc musi być większa niż 23.9, a jeśli chcesz załadować satelitę, ustal na tę właśnie prędkość. Na ekranie monitora zobaczysz linię w kształcie "S", która pokazuje pozycję satelity i promu na około Ziemi. Światełko normalne to pozycja promu, zaś migające to pozycja satelity. Ukażą się wtedy dwa radary. Radar po lewej stronie pokazuje twoje (Z, Y), a radar po prawej (X, mikro–Y). Za każdym razem jak tylko uda Ci się załadować satelitę, usłyszysz sygnał *RANDEZVOUS*.

Kolejność działań przy załadowaniu satelity:

Ustaw pozycję promu przez korekcję współrzędnych Z, Y i X w tym porządku, w jakim są podane (pamiętaj o tym, że "–" oznacza "z lewej" lub "na dole", zaś "+" oznacza "z prawej", bądź "na górze", "0" oznacza tę samą wysokość. Kiedy spotkasz satelitę, wszystkie współrzędne muszą być równe 0.

Zmiana orbity:

- utrzymuj współrzędną X aż do wysokości 210 mil,
- ustaw szybkość na 23.9 machów,
- naciśnij **R**, aby zaktywizować OMS,
- przestaw prom o 180 stopni – joy prawo lub lewo aby otrzymać YAW 180,
- ustaw PITCH na 004,
- ustaw szybkość do 19 machów,
- obróć prom dziobem w dół, poprzez zredukowanie YAW do zera.

Powrót:

Jest to jeden z najbardziej niebezpiecznych momentów lotu. W czasie wchodzenia w atmosferę prom rozgrzewa się do bardzo wysokiej temperatury. PITCH i wysokość muszą być stale kontrolowane, aby można było użyć systemu termicznego chronienia promu.

Eksploatacja energii obszaru końcowego:

W tej fazie lotu musisz dokładnie trzymać się trajektorii, którą masz na ekranie. Procedura, przez którą kontrolowana jest wysokość, rozpęd, szybkośc i kierunek jest nazwana TEAM.

Strata sygnalizacji:

W czasie wchodzenia promu w górne warstwy atmosfery wytwarzają się wokół promu bańki zjonizowanego powietrza, które blokują wszelką komunikację z Ziemią. Dzieje się to na wysokości około 160 mil. Jesteś wtedy zdany na samego siebie. Jednakże od czasu do czasu fale przedostają się i wtedy na Twoim radarze pokazuje się migający punkt. Wykorzystaj to do poprawienia swojej pozycji.

Ekran powrotu:

Na Twoim ekranie powrotu **X** oznacza zatrzymanie OMS, **T** oznacza *TEAM*, zaś **L** oznacza moment, w którym przejdziesz do właściwego lądowania. Mały kwadracik reprezentuje Ciebie. Kontroluj swoją pozycję przy pomocy joysticka.

Teraz już wszystko zależy tylko od Ciebie! Przyjemnego "szuflowania" życzy

TATAR.

ATARI NACZEJ

Joy Transmission

Kiedy używa się dwu różnych komputerów jednocześnie, np. ATARI i Commodore, często zachodzi potrzeba przesyłania danych pomiędzy nimi. Istnieje kilka sposobów rozwiązania tego problemu. Można próbować odczytać dyskietkę w formacie Commodore na stacji ATARI (lub odwrotnie), można próbować odczytać na magnetofonie Commodore program nagrany w formacie turbo ATARI. Ale po co? Projektanci obydwu komputerów wyposażyli je w kilka złącz, które mogą wysyłać i odbierać dane. Należą do nich:

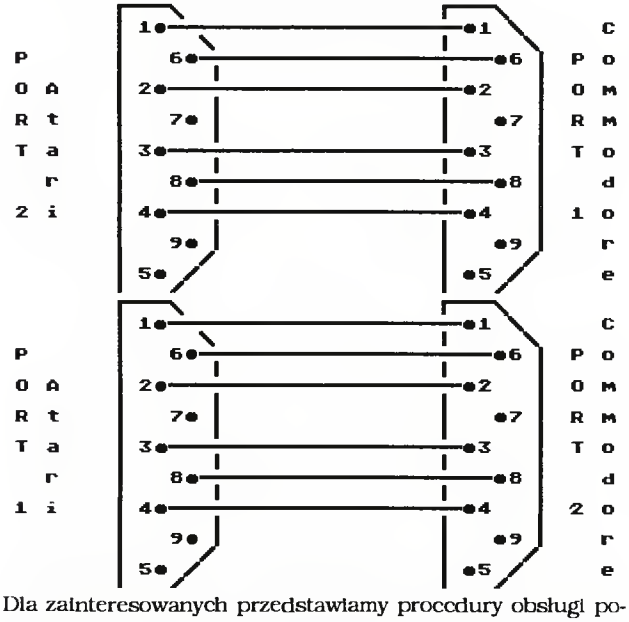
A) w Commodore:
- User Port (RS-232 i inne)
- Serial Bus (stacja(e) dysków, drukarka)
- Extension Port (cartridge, RAM(ROM)disk)
- Cassette (magnetofon)
- Control Ports (joystick, paddle, pióro świetlne, inne)

B) w ATARI:
- Peripheral (stacja dysków, magnetofon i inne)
- Parallel Bus (dysk twardy, inne)
- Cartridge
- Controllers (joystick, paddle, pióro świetlne, inne)

W obu komputerach jednakowym złączem jest *controller*. Te złącza postanowiliśmy wykorzystać. Porty joysticków są ze sobą w pełni zgodne. Obydwa komputery oferują możliwość pełnego programowania tych złącz, tzn. każdy z ośmiu bitów portu można ustawić jako wejściowy (0) lub wyjściowy (1). Nie nastręcza też żadnych kłopotów wykonanie odpowiedniego kabla połączeniowego. Wystarczy do tego zestaw następujących części:

- 4 wtyczki typu Eltra 9pin
- dwa kable sześcio-żyłowe.

Poniżej przedstawiamy schemat połączeń:



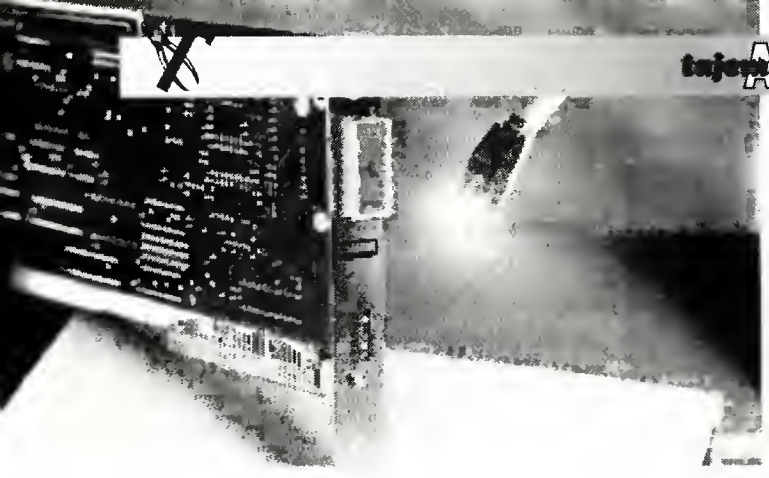
Dla zainteresowanych przedstawiamy procedury obsługi powyższego połączenia, napisane w języku maszynowym dla obu komputerów. Program źródłowy na Commodore został napisany przy użyciu programu Turbo Assembler 5.1, natomiast na ATARI programem Quick Assembler 1.0.

Procedury dla Atari:

```
*-- SYSTEM --

trigØ  equ $dØ1Ø
porta  equ $d3ØØ
pactl  equ $d3Ø2

*-- PROGRAM --

* odbieranie

       sei
       ldx #%ØØØØØØØ1
       jsr SetPort

       lda #Ø            ; w tym miejscu pętla
       sta porta         ; GetSgn skończy swe działanie

* procedura odbioru bajtu

       lda porta         ; ta pętla przestanie
Synch  cmp porta         ; działać, gdy wartość na
       beq Synch         ; porcie ulegnie zmianie (inc
ciaprb)

       lda #1            ; daj znak gotowości
       sta porta

       ldx #5Ø           ; odczekaj, żeby C64
       dex               ; odebrał ten znak
       bne *-1

       lda porta         ; wartość z portu
       and #%11111111Ø   ; dodaj do wartości
       ora trigØ         ; odczytanej z FIRE

       stx porta         ; nie pozwól, aby C64 przesłał
                         ; kolejny bajt bez twojej zgody

       jmp ResPort

* nadawanie

       sei
       ldx #%11111111    ; cały port na
       jsr SetPort       ; na wyjście

* procedura nadawania bajtu

       inc porta

Synchr ldx trigØ         ; czekaj na znak gotowości
       bne Synchr        ; od Commodore

       sta porta         ; zapisz do portu wysyłaną wartość

       ldx #Ø            ; odczekaj moment,
       inx               ; w którym C64 odbiera
       bne *-1           ; daną

ResPort equ *            ; KONIEC

       ldx #Ø
       stx porta

SetPort equ *            ; ustaw port

       lda pactl
       and #%111111Ø11
       sta pactl
       stx porta
       ora #%ØØØØØ1ØØ
       sta pactl

       cli
       rts
```

Procedury dla Commodore:

```
;*---- Zero Page

byte   = $Ø2             ;komórka pomocnicza

;*---- Registers

ciapra = $dcØØ           ;Rejestr Portu A
ciaprb = $dcØ1           ;Rejestr Portu B
ciddra = $dcØ2           ;Rejestr kierunku danych Portu A
ciddrb = $dcØ3           ;Rejestr kierunku danych Portu B

;*---- Program

;   WYSYŁANIE

       sei
       lda #%ØØØ1111Ø    ;Ø - wejście
       sta ciddra
       lda #%ØØØØ1111    ;1 - wyjście
       sta ciddrb

GetSgn lda ciapra        ;Dopóty bitØ zapalony
       lsr a             ;dopóki Atari niegotowe
       bcs GetSgn        ;do odbioru transmisji

;Powyższa pętla wykonywana jest tylko raz
;na samym początku programu.

;!!! Commodore zawsze zaczyna pierwszy !!!

;Tu umieszczona jest procedura wysyłania bajtu.
```



```
        tax             ;zachowaj akumulator

        inc ciaprb      ;daj znak, że zostanie
                        ;przesłany kolejny bajt

synch   lda ciapra      ;sprawdź, czy Atari
        lsr a           ;odebrało tę informację
        bcc synch       ;C=0 -> jescze nie

        txa             ;umieść starszy nybble
        lsr a           ;w porcie B
        lsr a
        lsr a
        lsr a
        sta ciaprb

        txa             ;przenieś bit0 bajtu
        and #$0f        ;do bitu4 (FIRE)
        lsr a
        bcc bit0        ;Tak spreparowany
                        ;młodszy nybble
        ora #%00001000  ;umieść w porcie A
                                ;(dla wygody Atari)
bit0    asl a
        sta ciapra

        ldx #0          ;daj czas Atari na
        inx             ;odebranie bajtu
        bpl *-1

        lda #%00000000  ;zainacjalizuj
        sta ciddrb      ;porty na standar-
        lda #%11111111  ;dowe wartości;
        sta ciddra      ;Port A wejściowy
        cli             ;Port B wyjściowy.
        rts

;  ODBIERANIE

        sei
        lda #%00010000
        sta ciddra
        sta ciapra
        lda #%00000000
        sta ciddrb

;Tu umieszczona jest procedura odbierania bajtu.

        lda ciapra      ;Jeśli na Porcie A zaszła
getbyt  cmp ciapra      ;zmiana, to Atari chce
        beq getbyt      ;przesłać kolejny bajt

        lda #%00000000  ;daj znak, żeś gotowy;
        sta ciapra      ;trig0 w Atari zeruje się

        ldx #50         ;poczekaj, aż Atari
        dex             ;odbierze ten znak
        bne *-1

        lda #%00010000  ;pewnie odebrało,
        sta ciapra      ;to skasuj go teraz

        lda ciaprb      ;w Porcie B
        asl a           ;starszy nybble
        asl a
        asl a
        asl a
        sta byte

        lda ciapra      ;w Porcie A
        and #$0f        ;młodszy nybble
        ora byte

        lda #%11111111  ;Przywróć standardową
        sta ciddra      ;wartość Portu A
        cli

        rts
```

Zamieszczamy także program **Joy Transmission**, wykorzystujący wyżej przedstawione procedury.

Po uruchomieniu programu na ATARI zgłosi się menu: **W** – wczytaj, **N** – nadaj, **Z** – zapisz, **O** – odbierz, **D** – numer dysku, **I** – informacje, **K** – katalog D1:, **ESC** – do widzenia i napis: **B=94b3 L=0000**

Dla programu uruchamianego na Commodore wystąpią drobne różnice w wyglądzie menu. Zamiast opcji **D** pojawi się **U** – urządzenie, czyli:

- 1 – magnetofon w systemie NORMAL
- 7 – magnetofon w systemie TURBO (jeśli użytkownik posiada cartridge z turbo)
- 8, 9 – stacja dyskietek o podanym numerze

Jeśli wystąpią jakiekolwiek problemy podczas transmisji danych, to można ją przerwać klawiszem RESET (dla Commodore – RESTORE), co spowoduje ponowne uruchomienie programu.

Podczas uruchamiania transmisji zawsze zaczyna Commodore, co jest sygnalizowane przez program.

W menu podane są także dwa parametry:

- **B** – wielkość bufora
- **L** – długość pliku znajdującego się w buforze

Dla ATARI parametr **B** jest zmienny i zależy od używanego DOS-a, dla DOS-a 2.5 z nakładką CP.SYS wynosi ona 9424. Dla Commodore wielkość **B** jest stała i wynosi C233. Wynika z tego, że długość bufora dla Commodore jest większa niż dla ATARI. Należy zatem uważać przy transmisji w kierunku Commodore -> ATARI na to, by długość programu (drugi parametr) była mniejsza, bądź równa wielkości bufora ATARI. W przeciwnym przypadku mogą wystąpić kolizje pamięci, co jest równoznaczne z zawieszeniem się ATARI.

Na ATARI kod wynikowy zgrywa się programem *Zgrywus+*.

Dla Commodore trzeba wykonać następujące czynności:

- 1. wykonać reset komputera, np. za pomocą komendy SYS 64738;
- 2. wykonać ciąg instrukcji:
  `POKE 43,1:POKE 44,32:POKE 8192,0:NEW`
- 3. wpisać zamieszczony program;
- 4. uruchomić program komendą RUN;
- 5. wykonać ciąg instrukcji: POKE 43,1:POKE 44,8:POKE 45,205:POKE 46,12 SAVE "JOYTRANS.OBJ",8,1

Zgrany w ten sposób program wczytujemy komendą LOAD "JOYTRANS.OBJ",8 i uruchamiamy komendą RUN.

Na koniec kilka uwag praktycznych:

- Przed połączeniem należy obydwa komputery wyłączyć.
- Najpierw należy włączyć ATARI, a później Commodore.
- Port 1 Commodore łączymy z portem 2 Atari, port 2 Commodore łączymy z portem 1 Atari.
- Połączenie i program są odporne na wszelkie zakłócenia, np. CB radio, odkurzacz przemysłowy, migoczącą świetlówkę, poskręcane przewody, bliskość monitora, spadki napięcia w sieci do 190V.
- Prędkość transmisji osiąga ok. 16 Kbodów.

```
IJ  1000 REM *--------------------------*
GW  1002 REM : ATARI<>C64 Transfer A    :
FJ  1004 REM :  autor:                  :
ZR  1006 REM : (c) 1993 Tajemnice ATARI :
JP  1008 REM *--------------------------*
KK  1010 DATA ffff00b7f0bba90c8d4203a200
KO  1020 DATA 2056e4a9038d4203a9548d4403
IB  1030 DATA a9ba8d4503a90c8d4a03a20020
AN  1040 DATA 56e4a201b50c9d35b7bd48ba95
IK  1050 DATA 0cca10f33003203 4b7a20020c5
BO  1060 DATA b978ca9a58a900388de502ede7
SM  1070 DATA 028582a9b78de602ede802aaca
YM  1080 DATA 8683a9048552a922855320a0b8
FO  1090 DATA 201fbaa03e2021ba200fbaa208
AH  1100 DATA dd2dbaf005ca10f830f18688a8
VU  1110 DATA 2083f9a2008e48038e4903b951
YR  1120 DATA fba82021ba201fbaa5880aaabd
HG  1130 DATA 37ba48bd36ba4860a201bd35b7
CE  1140 DATA 950cca10f86c0a0020e2b8a904
OP  1150 DATA 2057b8306da210a5829d4803a5
JG  1160 DATA 839d4903a9072073b8a210bd48
FC  1170 DATA 038d4ababd49038d4bba4c22b8
TO  1180 DATA 20e2b8a9082057b83041a210ad
```

```
IK  1190 DATA 4aba9d4803ad4bba9d4903a90b
WW  1200 DATA 2073b84c22b8a210a94d9d4403
XY  1210 DATA a9ba9d4503a9062065b83018a2
UI  1220 DATA 10a9009d48039d4903a907206a
NY  1230 DATA b83007a82021ba4c0ab8a908a2
PL  1240 DATA 1fa0bb20f7b9a58b203db8a210
QQ  1250 DATA a90c9d42032056e44c62b74829
CW  1260 DATA f04a4a4a4a204ab868290f1869
UE  1270 DATA 30c93a90026906a84c21ba48a2
AT  1280 DATA 10a9f19d4403a9bb9d4503689d
FE  1290 DATA 4a03a9039d42032056e4848b60
OL  1300 DATA 48ade7029d4403ade8029d4503
KI  1310 DATA 684c6ab8ae4ebae8e0399002a2
XC  1320 DATA 318e4eba4c5fb7a9b0a241a0bb
OI  1330 DATA 20f7b94c62b7a9bba257a0ba20
GE  1340 DATA f7b9a0b038ad4ebae92009158a5
HV  1350 DATA 82a683a0d220c4b8ad4abaae4b
QB  1360 DATA baa0d920c8b88a48290f20d3b8
ID  1370 DATA 684a4a4a4a186910c91a900269
CH  1380 DATA 06098091588860a90da212a0bb
OC  1390 DATA 20f7b9a2f1a0bb8e44038c4503
QR  1400 DATA a9008d4803a9108d4903a9058d
CE  1410 DATA 4203a2004c56e4ade7028586ad
XM  1420 DATA e8028587205fb978a2ff20c5b9
ZD  1430 DATA a001b94aba204bb98810f7ae4b
AK  1440 DATA bae88689a000b186204bb9c8d0
MK  1450 DATA f8e687c689d0f2a9ff8d00d3a2
HA  1460 DATA 0020c5b9584c62b7ee00d3ae10
WK  1470 DATA d0d0fb8d00d38d1ad0a200e810
TX  1480 DATA fd60a91aa227a0bb20f7b94c0f
GZ  1490 DATA baade7028586ade8028587205f
GO  1500 DATA b978a20120c5b9a9008d00d3a0
UT  1510 DATA 0220d6b99949ba88d0f7a482cc
CL  1520 DATA 4abaa6838aed4bbab0068e4bba
GZ  1530 DATA 8c4abaae4bbae88689a0002 0d6
CS  1540 DATA b99186c8d0f8e687c689d0f2a9
BX  1550 DATA 008d00d3a2002 0c5b9584c62b7
AK  1560 DATA ad02d329fb8d02d38e00d30904
DT  1570 DATA 8d02d360ad00d3cd00d3f0fba9
PZ  1580 DATA 018d00d3a232cad0fdad00d329
VK  1590 DATA fe0d10d08e00d38d1ad0608481
WB  1600 DATA 8680858aa0008489b180a82021
DP  1610 DATA baa489c8c48a90f160a0ff8cfc
TC  1620 DATA 02adfc02c9fff0f98cfc0260a0
OZ  1630 DATA 9ba200bd470348bd4603489860
VY  1640 DATA 1c2e17050823 0d3a0c9db7aab7
EW  1650 DATA d6b7f6b76ab908b993b883b85e
MF  1660 DATA b734b700000044313a2a2e2a9b
OC  1670 DATA 453a9b7dc1f4e1f2e9adc3efed
TI  1680 DATA edefe4eff2e5a0caeff9a0d4f2
NO  1690 DATA e1eef3ede9f4e9efee19572e77
BJ  1700 DATA 637a7974616a2070 6c696b204f
FQ  1710 DATA 2e6f64626965727a2064616e65
FX  1720 DATA 99195a2e7a617069737a2070 6c
FQ  1730 DATA 696b20204e2e6e6164616a2064
VK  1740 DATA 616e6520209919442e6e756d65
NY  1750 DATA 72206479736b752020492e696e
JY  1760 DATA 666f726d61636a6520209919 4b
LS  1770 DATA 2e6b6174616c6f672044203a20
NU  1780 DATA 204553432e646f7769647a656e
TZ  1790 DATA 696199a0e2bda0a0a0a0a0e4bd
QA  1800 DATA a0a0a0a0a0a8c3a9b1b9b9b3a0
ZE  1810 DATA c3a6c3a0f3efe6f46e617a7761
HP  1820 DATA 2070 6c696b753a9b9b73746174
IJ  1830 DATA 75733a4336342070 6965727773
DG  1840 DATA 7a799b52455455524e2d737461
HY  1850 DATA 72749b4175746f727a793a9b4b
QQ  1860 DATA 727a79737a746f662043687564
TE  1870 DATA 7a696e736b6920284174617269
BG  1880 DATA 299b4a616e75737a2044616272
MI  1890 DATA 6f77736b692028436f6d6d6f64
JV  1900 DATA 6f7265299b9b50726f64756365
OX  1910 DATA 6e7420692064797374727962 75
ZV  1920 DATA 746f723a9b43264320736f6674
XK  1930 DATA 9b55736c7567692 04b6f6d7075
ME  1940 DATA 7465726f77659b756c2e506f7a
JP  1950 DATA 6e616e736b612032619b33352d
IZ  1960 DATA 3038322052 7a65737a6f779b74
ZW  1970 DATA 656c2e2028302d313729343234
BW  1980 DATA 2d3030207 72e3330399be002e1
RW  1990 DATA 0200b7
```

```
0 REM JOYTRANSMITION BY JD
1 SUM=0 :OBJ=2049
2 DEF FN W(D)=(48-D)*(D<58)+(55-D)*(D>64)
3 FOR L=1000 TO 181 :PRINT L;CHR$(145) :READ D$
4 FOR D=1 TO LEN(D$) STEP 2 :B$=MID$(D$,D,2)
5 B=16*FN W(ASC(MID$(B$,1,1)))
6 B=B+FN W(ASC(MID$(B$,2,1)))
7 POKE OBJ,B :OBJ=OBJ+1 :SUM=SUM+B :NEXT D,L
8 IF SUM=131922 THEN END
9 PRINT "POPRAW DANE !" :STOP
100 DATA 0B08C9079E32303631000000A201BD
101 DATA 18039DCE0BBDCC0B9D1803CA10F120
102 DATA 2A0A2C01DC30036C1603789AA91B8D
103 DATA 11D058A980209 0FFA0008C20D08C21
104 DATA D0A233A9C2A02620C809ADCA0CA000
105 DATA 20D509AECB0CADCC0CA02D20C809A9
106 DATA E6A00A201EAB20E4FFC90DF0DFA208
107 DATA DDB30BF005CA10F830ED20D2FF2083
108 DATA 08A9ABA00B201EABF0DE8A0AAABDBC
109 DATA 0B48BDBB0B486020 9F09A000A901AE
110 DATA CA0C20BAFFA2CDA00CA900200D5FF90
111 DATA 0DA5BAC90890032 0E608A200A0CF38
112 DATA 8AE9CD8DCB0C98E90C8DCC0C6020 9F
113 DATA 09A00198AECA0C20BAFF18A9CD85AC
114 DATA 6DCB0CAAA90C85AD6DCC0CA8A9AC20
115 DATA D8FF90DB24903 0D7A5BA20B4FFA96F
116 DATA 2096FFA90D20D2FF20A5FFC90DD0F6
117 DATA 20ABFFA5BA20B1FFA96F2093FFA955
118 DATA 20A8FFA94920A8FF4CAEFFA901A2CB
119 DATA A00B20BDFFAECA0CE00890 97A901A0
120 DATA 6020BAFF20C0FF24903 01BA5BA20B4
121 DATA FFA5B92096FFA20420A5FFCAD0FAA9
122 DATA 0D20D2FF24905005A5B84CC3FF20A5
123 DATA FFAA20A5FF20CDBDE6D320A5FF20D2
124 DATA FFD0F8A202A5CBC93FF0DFB0CEC5CB
125 DATA F0FCD0C8A9D0A00B201EAB20E4FFC9
126 DATA 0DF014C931F008C9379 0F1C93AB0ED
127 DATA 20D2FF2909F8DCA0C60A9A7A00C201E
128 DATA ABA00C84D3A2FFE820CFFF9DB40CE0
129 DATA 15B004C90DD0F1A900 9DB40C8AA2B4
130 DATA A00C4CBDFF20CC098A484A4A4A4A20
131 DATA D5096829 0FC90A900269266930 968
132 DATA 0BC860A9E2A00B201EAB78A91E8D02
133 DATA DCA90F8D03DCA9CD85FBA90C85FCA9
134 DATA 3685 01AD00DC4AB0FAA002B9CA0C20
135 DATA 390A88D0F7AECC0CE886FDB1FB2039
136 DATA 0AC8D0F8E6FCC6FDD0F28C20D0A200
137 DATA 8E03DCCA8E02DCA9 3785 01586 0AAEE
138 DATA 01DC8D20D0AD00DC4A90FA8A4A4A4A
139 DATA 4A8D01DC8A2900F4A900209080A8D00
140 DATA DCA200E810FD60A9E2A00B201EAB78
141 DATA A9108D02DC8D00DCA9008D03DCA9CD
142 DATA 85FBA90C85FCA00220A60A99CA0C88
143 DATA D0F7AECC0CE886FD20A60A91FBC8D0
144 DATA F8E6FCC6FDD0F28C20D0A9FF8D02DC
145 DATA 5860AD00DCCD00DCF0FBA9008D00DC
146 DATA A232CAD0FDA9108D00DCAD01DC0A0A
147 DATA 0A0A8502AD00DC2900F05028D20D060
148 DATA A9F6A00B4C1EABA201BDCE0B9D1803
149 DATA CA10F76C02A09A080E9312C34F4D4D
150 DATA 4F444F52452DC15441524920CA4F59
151 DATA 20D452414E534D4954494F4E8DB5D7
152 DATA 2E57435A5954414A2050 4C494B20DA
153 DATA 2E5A415049535A2050 4C494B20B68D
154 DATA B5CF2E4F44424945525A2044414E45
155 DATA 20CE2E4E4144414A2044414E45202 0
156 DATA B68DB5CB2E4B4154414C4F472020 20
157 DATA 2020 0D52E55525A41445A454E4945
158 DATA 2338B68DB5C92E494E464F524D4143
159 DATA 4A4120 20205F2E444F5749445A454E
160 DATA 49412020B68D12423DA4A4A4A42044
161 DATA 3DA4A4A4A4202028C3293139393320
162 DATA C326C320534F46548D8D3E1220929D
163 DATA 00575A4B554E4F495F8E08C1081909
164 DATA 7B09E3095F0AD00AD70A241E080000
165 DATA 8DCE554D45522028312C372D392912
166 DATA 209D008DC35A454B414D204E4120C1
167 DATA 544152492E2E2E008DC155544F525A
168 DATA 593A8DCB525A59535A544F4620C348
169 DATA 55445A494E534B492028C154415249
170 DATA 298DCA414E55535A20C44142524F57
171 DATA 534B492028C34F4D4D4F444F524529
172 DATA 8D8DD0524F445543454E5420492044
173 DATA 595354525942555544F523A8DC326C3
174 DATA 20534F46548DD5534C55474920CB4F
175 DATA 4D50555445524F57458D554C2ED04F
176 DATA 5A4E414E534B41203241 8D33352D30
177 DATA 38322 0D25A45535A4F578D54454C2E
178 DATA 20283 0D3137293432342D30302057
179 DATA 2E333039008DCE415A57412050 4C49
180 DATA 4B553A00000000000000000000000000
181 DATA 00000000000000000000000800000
```

Janusz Dąbrowski
Krzysztof Chudziński
"C&C soft"

O MAGNETOFONIE

Wielu użytkowników magnetofonów nie zdaje sobie sprawy z tego, że ilość błędów transmisji związana jest z czasem pracy magnetofonu bez konserwacji. Chciałbym opisać tutaj najczęściej występujące niedomagania magnetofonu, a także sposoby ich likwidacji.

Głowica.

Odbiera sygnały z taśmy i zamienia je na impulsy elektryczne. W przypadku, gdy jest zabrudzona, sygnał jest słabszy i mniej dokładny, co powoduje częste występowanie błędów. Aby wyczyścić głowicę, potrzebna jest zapałka, odrobina waty i płyn do czyszczenia głowic, który można kupić w sklepach z częściami elektronicznymi. Gdy takiego płynu nie ma, można użyć zamiennie czystego spirytusu. Mocno owijamy watkę na zapałce tak, aby się nie kudłaczyła i nie zostawiała włosków, zamaczamy ją delikatnie w spirytusie i delikatnie przecieramy głowicę. Po przetarciu dobrze jest wytrzeć głowicę do sucha innym kawałkiem waty, aby nie pozostały na niej zacieki. Jednak nie zawsze wystarczy samo wytarcie głowicy. Bywa często tak, że mimo tego, iż głowica jest czysta, sygnał jest nadal słaby. Przyczyną tego może być inne ustawienie głowicy w magnetofonie, z którego odczytujemy niż w tym, na którym kaseta była nagrywana. W tym przypadku nie ma żadnej żelaznej zasady, jak ustawić głowicę, by odczytywała wszystkie kasety. Do każdej z kaset, które się nie dają odczytać z powodu nieprawidłowo ustawionej głowicy trzeba będzie prawdopodobnie ustawiać ją osobno. Należy przy tym pamiętać, że każde ustawianie głowicy powoduje luzowanie się mocowania śrubki do regulacji i przy częstych tego typu zabiegach może doprowadzić do tego, że głowica nie da się ustawić. Należy wtedy poszukać wkrętu z gwintem grubszym o 0.1 mm od poprzedniego i spróbować go wkręcić. Głowicę regulujemy delikatnie pokręcając lewą (patrząc od strony klawiszy) śrubką. Dobrze jest zaznaczyć sobie położenie początkowe (przed regulacją) i odliczać obroty śrubki od tego położenia po to, aby w przypadku nieudanej regulacji można było wrócić do położenia początkowego. W sytuacji, gdy ani czyszczenie ani regulacja (w przypadku gdy kaseta jest na pewno dobra i daje się odczytać na innym magnetofonie) nie pomagają, winę może ponosić stara, zużyta głowica. Sposobu wymiany głowicy nie będę opisywał, ponieważ zaawansowani będą potrafili to zrobić, a początkującym proponuję zwrócenie się z tą sprawą do usługowego warsztatu elektronicznego.

Rolka dociskowa.

Zrobiona jest z twardej gumy, a jej zadaniem jest równomierne przesuwanie taśmy. W przypadku, gdy jest uszkodzona lub zabrudzona słychać podczas transmisji charakterystyczne wibracje sygnału. Często wystarcza wyczyszczenie jej takim samym sposobem jak głowicę, czasem jednak istnieje potrzeba wymiany rolki na nową. Rolkę taką można dostać w sklepach elekronicznych. Aby włączyć przesuw taśmy na czas czyszczenia rolki (jest to bardzo pomocne) należy napisać POKE 54018,52. Wyłączenia przesuwu następuje po wpisaniu POKE 54018,60.

Styki.

Występują we wtyczce, a ich zabrudzenie (śniedzieją) objawia się trzaskami podczas transmisji. Styki należy czyścić okresowo co dwa, trzy miesiące przy częstym korzystaniu z magnetofonu. Najlepiej używać do tego bardzo drobnego papieru ściernego. Styki dają się łatwo wyciągnąć po wciśnięciu śrubokrętem każdego z nich od zewnętrznej strony. Styki najlepiej wyciągać i czyścić pojedynczo, aby uniknąć pomieszania przewodów. Styki znajdują się również w środku magnetofonu przy klawiszach służących do włączania przewijania taśmy. Często po dłuższej eksploatacji obrywają się, co powoduje, że nie można bez specjalnych zabiegów przewijać taśmy. W takim wypadku najlepiej złutować końcówki i dzięki temu zasilanie będzie cały czas w układzie i będzie można przewijać.

Sprężynka.

Jej zadaniem jest dociskanie rolki do taśmy. Gdy jest za mocno napięta, powoduje wyginanie wałka napędzającego i przez to nierównomierny przesuw taśmy. Gdy sprężynka jest za luźna, to taśma przesuwa się z nieodpowiednią prędkością. W przypadku zniszczenia sprężynki jej zadanie dobrze spełnia sprężynka od długopisu. Trzeba ją odpowiednio przyciąć i założyć w miejsce poprzedniej.

Kaseta.

Nie powinna leżeć w zawilgoconym miejscu ani być narażona na działanie wysokich czy bardzo niskich temperatur. Kaset nie należy pożyczać, gdyż każdy magnetofon inaczej traktuje taśmę i nie wiadomo czy po powrocie będzie się ona jeszcze nadawała do użycia.

Kilka drobnych uwag.

Zaraz po zakończeniu odczytu z taśmy magnetofon należy wyłączyć. Zapobiegnie to powstaniu rowka w miejscu, gdzie rolka dociska taśmę do wałka napędowego. Gdy taki rowek powstanie, jest mało prawdopodobne, że dane z drugiej strony kasety nagrane w tym miejscu dadzą się odczytać. Może czasami zaistnieć sytuacja, że mimo sprawnych podzespołów plik nie daje się odczytać. Winę za to może ponosić klapka w magnetofonie, która w większości przypadków nie zamyka się do końca, a przez to źle ustawia taśmę w stosunku do wałka i głowicy. Czasami wystarczy położyć coś ciężkiego na klapce, a czasami trzeba klapkę całkowicie zdjąć. Jest to niezbyt estetyczne, natomiast bardzo wygodne przy częstych wymianach kaset.

Życzę wszystkim, aby ich magnetofony psuły się jak najrzadziej, a programy wczytywały jak najczęściej.

Marek Pasek

Przesyłam garść poprawek do kilku ciekawych gier do rubryki "Z pamiętnika łamacza joysticka":

- **DIGI DUCK** – wystukaj na planszy tytułowej gry hasło **"DE JET"**, a otrzymasz 99 żyć.
- **HANS KLOSS** – wystukanie na planszy tytułowej napisu **"HEAVY METAL"** daje nieskończoną ilość jedzenia oraz napojów.
- **MASTER HEAD** – na stronie tytułowej wpisanie tekstu **"SILESIA"** przenosi na 16 poziom, a napis **"SCORPIO"** przenosi na 31 poziom.
- **OPERATION BLOOD** – na planszy tytułowej należy napisać **"BC"**; podczas gry działa kombinacja klawiszy **SELECT+OPTION+CONTROL** w połączeniu z:
  ESC – dodatkowe granaty,
  <strzałka w górę> – dodatkowe magazynki,
  <przecinek> – energia,
  T – następny poziom.
- **TOMAHAWK** – naciśnij klawisz **M** – mapa; klawiszami **5, 6, 7, 8** będziesz się mógł przesuwać, ale tylko stojąc na lotnisku.
- **BUBBLE BOBBLE** – gdy tracisz ostatnie życie naciśnij i przytrzymaj **FIRE**.

Berenika Lenart, Kędzierzyn–Koźle

Od redakcji: dziękujemy za poprawki i prosimy o więcej. Co do Twoich pytań, to temat TURBO 2000 plus jeszcze powróci i to już niedługo.

Wykaz punktów sprzedających bieżące i archiwalne numery Tajemnic Atari. Drukiem pogrubionym zaznaczono miejsca, w których można równocześnie kupić legalne oprogramowanie L.K. AVALON

PHU 'FAMPOL'
'CIDER CENTRUM'
Strzelców Bytomskich
Bytom Stroszek
MIKRA, Jesionowa 3
Chorzów
DRZAJA
Mirowska 6, Częstochowa
INFO PANDA
Śniadeckich 20, 50, Bydgoszcz
SPRZĘT KOMPUTEROWY
Kościuszki 6/26, Goleniów
MICROMAN
Plebiscytowa 31, Katowice
Rynek 4, Rybnik
DH HERMES, Rybnik
SUPERMARKET, Sosnowiec
Wojska Polskiego 14,
Bielsko-Biała
KRYMARYS
Toruńska 63, Koło
KSIĘGARNIA TECHNICZNA
Podwale 4, Kraków

BIT - PLUS
Bieżanowska 1, Kraków
VIDEO CENTRUM
Kościelna 5, Krynica Górska
RETURN Studio Komputerowe
Zamojska 25, Lublin
POMAREX
Bernardyńska 20,
Niecała 12, Lublin
OKW Elektronika
Piotrowska 39, Łódź
Firma Handlowo Usługowa BD
Rynek 35, Nowy Targ
MEGABAJT
Reymonta 23, Opole
TRACK WS
Kołłątaja 11, Opole
PC KOMPUTER
Ogrody 11a, Ostrowiec Św.
BI&K COMPUTER
Wierzbięcice 37, Poznań
P.H. METRO
Ratajczaka 31, Poznań

STUDIO-SIX
Rejtana 3, Przemyśl
BAJT, Rejtana 33, Rzeszów
DABI, Szopena, Rzeszów
DOMAR
Zygmuntowska, Rzeszów
Okulickiego, Rzeszów
STEL - DOM
Wojska Poiskiego 16, Słupsk
JOKER
Mickiewicza 15, St. Wola
KWANT Agencja Handlowa
Żydowska 6, Tarnów
SONIX Sp. z o. o.
Rapacklego 3
DT Centrum
Rydla (Helios 2), Szczecin
COMPUTER CENTER WARS
Marszałkowska 104/122
Warszawa
ATARI STUDIO
Gen. Abrahama 4, Warszawa

O.K.W. ELEKTRONIKA
Mokotowska 51/53, Warszawa
OSKAR
Grochowska 207 (Universam)
Jana Pawła II 58 (Maxim),
Igańska 26, Warszawa
DOMAINSOFTWARE
Składnica Harcerska przy
SDH FENIX, ul. Kurzy Targ 3/4
Wrocław
COMAT
Kniaziewicza 19, Wrocław
KSIĘGARNIA STAROMIEJSKA
Żeromskiego 16, Zielona Góra

Zachęcamy właścicieli sklepów zainteresowanych sprzedażą pisma lub oprogramowania do kontaktu z L.K. AVALON. Najszybszy kontakt telefoniczny pod numer (17) 62-74-71 wew. 274, 275 od 7:30 do 15:00. Oferujemy korzystne warunki współpracy. Firmy bądź sklepy zainteresowane kolportażem gazet otrzymują je na zasadach komisowych, a ponadto ich adresy są drukowane w TA, co stanowi dobrą reklamę.

L.K. AVALON
skr. poczt. 46, Strzyżów
proponuje:

Akcesoria

filtr monitorowy szklany, powlekany ołowiem	300.000
pokrywy na ATARI XL lub XE	50.000
taśma do drukarek STAR LC 10/20	60.000
MICROPRINT (prod. USA) umożliwiający podłączenie do ATARI dowolnej drukarki (CENTRONICS)	220.000
dyskietka czyszcząca 5.25' z płynem	50.000
kaseta czyszcząca z płynem	30.000

pudełka na dyskietki o pojemności:

5 szt.	30.000
10 szt.	35.000
50 szt.	100.000
100 szt.	120.000

Literatura

Leksykon komp. dla początkujących	30.000
Słownik term. i komunikatów komp.	30.000
Poznajemy komputer Atari XL/XE	30.000
Atari Basic dla dzieci	35.000
Logo Atari	35.000
Opisy gier na małe Atari	35.000
Elementarz ATARI	45.000
Atari Logo	55.000

Do łącznego zamówienia należy doliczyć koszt wysyłki (15.000 zł). W przypadku zamówienia również oprogramowania z oferty wewnątrz pisma, koszt wysyłki nie jest doliczany. Wszystkie ceny ważne są do ukazania się następnego numeru TA.

- Uniwersalny moduł cartridge do komputerów ATARI XL/XE
- Można na nim wielokrotnie zapisywać dowolne programy
- Po wyjęciu z komputera programy pozostają w module dzięki podtrzymaniu bateryjnemu
- Dla zaawansowanych symulacja napędu dyskietek

Do każdego RAM CARTu dołączone są programy pomocnicze na kasecie i dyskietce, oraz podręcznik użytkownika systemu opracowany w L.K. AVALON.

DODATKOWO, ZA DARMO nagrywamy na RAM-CART zestawy programów z TA (najlepsze gry z zestawów TA 1,2,3,4,5,6). Jedyna okazja zdobycia 64 lub 128 KB legalnego oprogramowania ZA DARMO!!!

Jako autoryzowany dystrybutor RAM CARTu udzielamy rocznej gwarancji działania modułu. W wypadku wystąpienia wad uniemożliwiających używanie systemu, drogą pocztową wymieniamy wadliwe egzemplarze na poprawnie działające.

Cena systemu z 64KB pamięci - 470.000 zł.
Cena systemu z 128KB pamięci - 750.000 zł.

L.K. AVALON, skr. poczt. 46, 38-100 Strzyżów

Uregulowanie należności następuje przy odbiorze przesyłki.

UWAGA!!!
OBNIŻKA CEN NA NIEKTÓRE GRY *AVALONU* - ZAGLĄDNIJ NA STR. 2!